S50728

(12)特許協力条約に基づいて公開された国際出願

(19) 世界知的所有権機関
国際事務局

(43) 国際公開日
2004 年10 月21 日 (21.10.2004)

PCT

(10) 国際公開番号
WO 2004/090712 A1

(51) 国際特許分類[7]: G06F 3/14, G09G 5/14

(21) 国際出願番号: PCT/JP2004/004414

(22) 国際出願日: 2004 年3 月29 日 (29.03.2004)

(25) 国際出願の言語: 日本語

(26) 国際公開の言語: 日本語

(30) 優先権データ:
特願2003-106393 2003 年4 月10 日 (10.04.2003) JP

(71) 出願人 (米国を除く全ての指定国について): 松下電器産業株式会社 (MATSUSHITA ELECTRIC INDUSTRIAL CO., LTD.) [JP/JP]; 〒5718501 大阪府門真市大字門真１００６ Osaka (JP).

(72) 発明者; および
(75) 発明者/出願人 (米国についてのみ): 灘本 雄二 (NADAMOTO, Yuji).

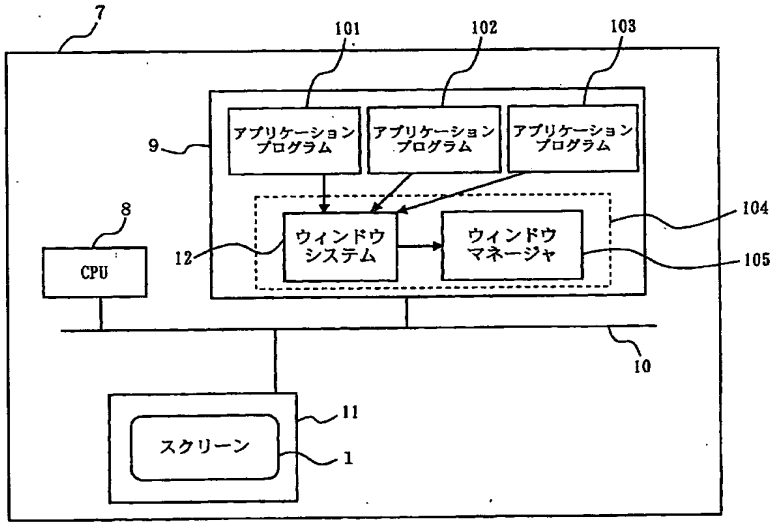
(74) 代理人: 小笠原 史朗 (OGASAWARA, Shiro); 〒5640053 大阪府吹田市江の木町３番１１号 第３ロンヂェビル Osaka (JP).

(81) 指定国 (表示のない限り、全ての種類の国内保護が可能): AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NA, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RU, SC, SD, SE, SG, SK, SL, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, YU, ZA, ZM, ZW.

[続葉有]

(54) Title: WINDOW STACK CONTROL METHOD, WINDOW MANAGEMENT PROGRAM, AND WINDOW MANAGEMENT APPARATUS

(54) 発明の名称: ウィンドウスタック制御方法、ウィンドウ管理プログラム、ウィンドウ管理装置



1O1...APPLICATION PROGRAM
102...APPLICATION PROGRAM
103...APPLICATION PROGRAM
12...WINDOW SYSTEM
105...WINDOW MANAGER
1...SCREEN

(57) Abstract: When controlling a window stack that manages a stack of windows during displaying, on a display device (11), a plurality of windows from a computer (7), application programs (101,102,103) specify groups for the windows, and a window management program (104) manages that the stack order of the windows is series in each group. This prevents a window display effected by an application displaying a window on the top from being accidentally influenced by another application.

(57) 要約: コンピュータ（７）から表示装置（１１）に複数のウィンドウを表示する際のウィンドウの重ね合わせを管理するウィンドウスタックを制御するに際し、アプリケーションプログラム（１０１，１０２，１０３）がウィンドウにグループを指定し、ウィンドウ管理プログラム（１０４）がウィンドウのスタック順をグループ毎に一連とする。これにより、最前面でウィンドウを表示中のアプリケーションのウィンドウ表示に、他のアプリケーションが不用意に影響を及ぼす事が無くなる。

WO 2004/090712 A1

(84) 指定国 (表示のない限り、全ての種類の広域保護が可能): ARIPO (BW, GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZM, ZW), ユーラシア (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), ヨーロッパ (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IT, LU, MC, NL, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

添付公開書類:
— 国際調査報告書
— 補正書

明細書

ウィンドウスタック制御方法、ウィンドウ管理プログラム、ウィンドウ管理装置

技術分野

本発明は、コンピュータから表示装置に複数のウィンドウを表示する際のウィンドウの重ね合わせを管理するウィンドウスタック制御方法に関する。

背景技術

近年、表示装置に複数のウィンドウを表示するコンピュータとして、ワークステーションや家庭用コンピュータが広く普及している。これらの場合、ユーザは、表示装置として比較的大きな画面に複数のウィンドウを表示して、マウスとキーボードを使ってウィンドウの重なりの上下関係であるスタック順の操作を行うのが一般的である。

例えば、Ｕｎｉｘ（Ｒ）ワークステーションで広く普及しているＸ１１ウィンドウシステムの場合、ウィンドウの重なりの上下関係であるスタック順は、ウィンドウの生成時にスタック最上位に位置付けられ、マップされた後は、マウスやアプリケーションの操作によって、スタック位置を上下方向に移動させる。

具体例として、コンピュータ上で動作する３種類のアプリケーションが合計５枚のウィンドウを表示装置のスクリーンに表示する際の表示例となる模式図を図２９に示す。図２９において、１はＣＲＴモニタや液晶モニタで実現さ

れる表示装置のスクリーン、２及び３は第１のアプリケーションが表示するウィンドウ、４及び５は第２のアプリケーションが表示するウィンドウ、６は第３のアプリケーションが表示するウィンドウであり、第１のアプリケーションが表示するウィンドウ２及びウィンドウ３を第１のグループ、第２のアプリケーションが表示するウィンドウ４及びウィンドウ５を第２のグループ、第３のアプリケーションが表示するウィンドウ６を第３のグループとする。

図２９を見ればわかるように、ウィンドウ３はウィンドウ２の上に重なり、ウィンドウ５はウィンドウ４の上に重なっている。この時のウィンドウスタック順の例となる模式図を図３０に示す。図３０を見ればわかるように、ウィンドウ３はウィンドウ２よりスタック上位にあるので、両者が表示上で重なる場合は、ウィンドウ３がウィンドウ２の上に重なる事となり、同様に、ウィンドウ５はウィンドウ４よりスタック上位にあるので、両者が表示上で重なる場合は、ウィンドウ５がウィンドウ４の上に重なって表示される。

そして、このような表示状態、スタック順状態で、第１のアプリケーションが何らかの都合で、例えば、ウィンドウ２をウィンドウ３の前面に表示させたくなった場合、ウィンドウ２をスタック最上位に持ってくる事で対応していた。この時のスタック順の模式図を図３１に示す。図３１を見ればわかるように、ウィンドウ２はウィンドウ３よりスタック上位にあるので、両者が表示上で重なる場合は、ウィンドウ２がウィンドウ３の上に重なって表示されるこ

ととなる。また、この操作により、ウィンドウ２が、第２のグループ、第３のグループのウィンドウを跨って、スタック順で最下位から最上位へ移動した事がわかる。

発明の開示

しかしながら、上記従来のウィンドウスタック制御方法においては、アプリケーションが或るウィンドウに対して、ウィンドウスタック移動、ウィンドウ生成及び表示を行った場合、グループを跨ってスタック制御をするため、意図せず、該ウィンドウが他のアプリケーションの表示するウィンドウの上に重なることがあるという問題があった。

上記課題に鑑み、本発明は、ウィンドウスタックの制御方法において、最前面でウィンドウを表示中のアプリケーションのウィンドウ表示に、他のアプリケーションが不用意に影響を及ぼす事の無いウィンドウスタック制御方法を提供することを目的とする。

この課題を解決するために本発明は以下の構成を採用した。なお、括弧内の参照符号および図番号は、本発明の理解を助けるために図面との対応関係を示したものであって、本発明の範囲を何ら限定するものではない。

本発明の第１の発明は、１つ以上のアプリケーションプログラム（１０１，１０２，１０３）に基づいて表示装置（１１）に表示される複数のウィンドウの重ね合わせを管理するためのウィンドウスタック制御方法であって、アプリケーションプログラムからウィンドウのグループの指定を受けるステップと、アプリケーションプログラムからウ

ィンドウの表示要求を受けるステップと、表示要求を受けて当該ウィンドウの表示を行う際に、当該ウィンドウのスタック順をグループ毎に一連となるように纏めるステップ（Ｓ３０８，図４，Ｓ２５１０，図２６）とを備えたウィンドウスタック制御方法である。

これにより、スタック順がグループ単位で纏められるため、最上位のグループに属する各ウィンドウは、他のグループのウィンドウの下に重ねられて表示される事が無くなり、最前面でウィンドウを表示中のアプリケーションのウィンドウ表示に、他のアプリケーションが不用意に影響を及ぼす事を無くす事ができる。特に、画面が小さく、結果として、ウィンドウの重なり易い携帯電話やＰＤＡ等で、その効果は顕著である。

また、本発明の第２の発明は、第１の発明において、前記纏めるステップは、グループ毎に纏めていない第１のウィンドウの表示要求をアプリケーションプログラムから受け取った際に、グループ毎に纏めていないウィンドウの中で前記第１のウィンドウと同じグループに属する第１のウィンドウ群のスタック順が、該第１のウィンドウ群の中でのスタック順関係は保持したまま、既にグループ毎に纏められたウィンドウの中で前記第１のウィンドウと同じグループに属する第２のウィンドウ群のスタック順と一連となるようにする事を特徴とするウィンドウスタック制御方法である（図２６）。

これにより、表示要求によって同一グループ内でみたスタック順が変更されることがないので、アプリケーション

の同一グループ内のスタック順管理の負担が軽減される。

また、本発明の第３の発明は、第１又は第２の発明において、前記ウィンドウスタック制御方法は、グループ毎に代表ウィンドウを１枚生成するステップをさらに備え、前記纏めるステップは、ウィンドウのスタック順をグループ毎に一連とする際に、当該ウィンドウを前記代表ウィンドウの子ウィンドウとする事を特徴とするウィンドウスタック制御方法である（図２２）。

これにより、グループ単位のスタック移動やグループ単位のウィンドウの表示／非表示の切り替えを代表ウィンドウに対する制御で実施できる分、グループ内のウィンドウが多いほど、処理負荷を軽減することができるようになる。

また、本発明の第４の発明は、第１から第３のいずれかの発明において、アプリケーションプログラムからグループ内でのウィンドウスタックの最上位移動要求および最下位移動要求を受けるステップと、前記最上位移動要求または前記最下位移動要求を受けて、グループ内でのスタック変更を行うステップ（Ｓ７０８，Ｓ７１０，Ｓ７１１，Ｓ１２０８，Ｓ１２１０，Ｓ１２１１）とをさらに備えたウィンドウスタック制御方法である。

これにより、スタックの操作がグループ内で閉じるので、各アプリケーションプログラムは、他のアプリケーションプログラムのウィンドウ表示を気にする事無くスタック変更を行う事ができる。

また、本発明の第５の発明は、第４の発明において、前

記スタック変更を行うステップは、前記第１のウィンドウに対するグループ内でのウィンドウスタックの最上位移動要求および最下位移動要求をアプリケーションプログラムから受け取った際に、グループ毎に纏めていないウィンドウの中で前記第１のウィンドウと同じグループに属する第１のウィンドウ群のスタック順が、前記第１のウィンドウ群の中でのスタック順関係は保持したまま、既にグループ毎に纏められたウィンドウの中で前記第１のウィンドウと同じグループに属する第２のウィンドウ群のスタック順と一連となるようにスタック変更を行うことを特徴とする（Ｓ７０８，図９，Ｓ１２０８，図１４）ウィンドウスタック制御方法である。

これにより、スタック変更対象となるウィンドウと同じグループに属する通常ウィンドウ全てが纏められるため、アプリケーションプログラムは、自身が設定したグループのウィンドウのスタック順を把握し易くなる。

また、本発明の第６の発明は、第１から第３のいずれかの発明において、アプリケーションプログラムからグループ単位でのウィンドウスタックの最上位移動要求および最下位移動要求を受けるステップと、前記最上位移動要求または前記最下位移動要求を受けて、グループ単位でのスタック変更を行うステップ（Ｓ１７０５，Ｓ１７０６，Ｓ２００４）とをさらに備えたウィンドウスタック制御方法である。

これにより、ウィンドウの操作をグループ指定で実施できるので、例えば、グループをアプリケーションプログラ

ム単位で振り分けた場合だと、そのアプリケーションに属するウィンドウのメンバーを知らなくても、スタック最前面にそのアプリケーションに属する全通常ウィンドウを移動させる事が可能となる。

また、本発明の第７の発明は、第６の発明において、前記スタック変更を行うステップは、第１のグループに対するグループ単位でのウィンドウスタックの最上位移動要求および最下位移動要求をアプリケーションプログラムから受け取った際に、グループ毎に纏めていないウィンドウの中で前記第１のグループに属する第１のウィンドウ群のスタック順が、前記第１のウィンドウ群の中でのスタック順関係は保持したまま、既にグループ毎に纏められたウィンドウの中で前記第１のグループに属する第２のウィンドウ群のスタック順と一連となるようにスタック変更を行うことを特徴とする（Ｓ１７０５，図１８，Ｓ２００４，図２１）ウィンドウスタック制御方法である。

これにより、スタック変更対象となるグループに属する通常ウィンドウ全てを、その中でのスタック順は保持したままで纏めた上でスタック変更を実施するので、アプリケーションプログラムは、自身が設定したグループのウィンドウのスタック順を把握し易くなる。

また、本発明の第８の発明は、第１の発明において、アプリケーションプログラムによって優先ウィンドウとして指定されたウィンドウをいずれのグループにも所属させず、かつ当該優先ウィンドウを、表示装置に表示された、いずれかのグループに所属する全てのウィンドウより常にス

タック上位に位置させることを特徴とするウィンドウスタック制御方法である。

これにより、グループ単位のスタック順に関係無くウィンドウを表示できるので、最優先で表示するウィンドウのサービスを提供することができる。

また、本発明の第９の発明は、第１の発明において、最上位グループに属するウィンドウよりスタック下位のウィンドウを常に非表示状態とすることを特徴とするウィンドウスタック制御方法である。

これにより、最上位グループよりスタック下位のグループが何を表示しようとしても表示装置には表示されないので、最上位グループのウィンドウで覆われない部分に中途半端に映り込む心配が無い。

また、本発明の第１０の発明は、第１の発明において、最上位グループに属するウィンドウの中のスタック最下位のウィンドウの直下に、特定のウィンドウを位置させることを特徴とする（図２３，図２４）ウィンドウスタック制御方法である。

これにより、例えば、前記特定のウィンドウを表示装置のスクリーン一杯に表示しておけば、前記特定のウィンドウのスタック下位のグループが何を表示しようとしても，表示装置に表示されないので、最上位グループのウィンドウで覆われない部分に中途半端に映り込む心配が無い。

また、本発明の第１１の発明は、第４又は第６の発明において、前記ウィンドウスタック制御方法は、Ｘウィンドウシステムとウィンドウマネージャによって複数のウィン

ドウの重ね合わせを管理するものであって、最上位グループの直上に特定のウィンドウを位置させることを特徴とするウィンドウスタック制御方法である。

これにより、最上位グループを別のグループに入れ替える際、Xウィンドウシステムの場合、XRestackWindows()関数を使用する事となるが、ウィンドウ群のスタック移動先を示すために使用できる。もし、最上位グループのウィンドウよりスタック上位にウィンドウが存在しなかった場合、存在する全てのウィンドウの並び順を決め、前記XRestackWindows()の引数に代入せねばならないが、前記特定のウィンドウがあれば、前記特定のウィンドウと前記ウィンドウ群を引数に代入するだけで済む。

また、本発明の第12の発明は、第1から第3のいずれかの発明において、前記ウィンドウスタック制御方法は、Xウィンドウシステムとウィンドウマネージャによって複数のウィンドウの重ね合わせを管理するものであって、ウィンドウマネージャが、ウィンドウ破棄通知受信時に、ウィンドウシステム間とでスタック認識の整合性を確認し、異なっている場合は、ウィンドウシステムのスタック認識をウィンドウマネージャのスタック認識に合わせ込む処理を行うことを特徴とするウィンドウスタック制御方法である。

これにより、ウィンドウ破棄通知受信時に整合性確保を図るので、ウィンドウシステムがスタック変更処理に失敗しても、処理が破綻する事がなくなる。

また、本発明の第１３の発明は、第１から第３のいずれかの発明において、前記ウィンドウスタック制御方法は、Ｘウィンドウシステムとウィンドウマネージャによって複数のウィンドウの重ね合わせを管理するものであって、ウィンドウマネージャが、ウィンドウシステムにスタック変更を要求する際にフラグを立てる一方で、ウィンドウ破棄通知受信時に、前記フラグが立っている時のみ、ウィンドウシステム間とでスタック認識の整合性を確認し、異なっている場合は、ウィンドウシステムのスタック認識をウィンドウマネージャのスタック認識に合わせ込む処理を行い、前記フラグを下げることを特徴とするウィンドウスタック制御方法である。

これにより、ウィンドウマネージャがスタック変更を実施した時のみ、スタック認識の整合性のための処理を行うので、処理の実行頻度を軽減できる。

また、本発明の第１４の発明は、第１の発明において、前記纏めるステップは、グループ毎に纏めていない第１のウィンドウの表示要求をアプリケーションプログラムから受け取った際に、既にグループ毎に纏められたウィンドウの中で前記第１のウィンドウと同じグループに属する第２のウィンドウ群のスタック順と一連となるようにすることを特徴とするウィンドウスタック制御方法である（図４）。

また、本発明の第１５の発明は、１つ以上のアプリケーションプログラム（１０１，１０２，１０３）に基づいて表示装置（１１）に表示される複数のウィンドウの重ね合

わせを管理するためのウィンドウ管理プログラム（１０４）であって、コンピュータ（７）に、アプリケーションプログラムからウィンドウのグループの指定を受けるステップと、アプリケーションプログラムからウィンドウの表示要求を受けるステップと、表示要求を受けた際に該ウィンドウのスタック順をグループ毎に一連となるように纏めるステップ（Ｓ３０８，図４）とを実行させるためのウィンドウ管理プログラムである。

また、本発明の第１６の発明は、表示装置（１１）に複数のウィンドウを表示する際のウィンドウの重ね合わせを管理するウィンドウ管理装置であって、１つ以上のウィンドウを前記表示装置に表示する１つ以上のアプリケーションプログラム（１０１，１０２，１０３）と、前記１つ以上のアプリケーションプログラムが表示するウィンドウの重ね合わせを管理するウィンドウ管理プログラム（１０４）と、前記アプリケーションプログラムおよび前記ウィンドウ管理プログラムを実行する処理部（８）とを備え、前記アプリケーションプログラムは、前記ウィンドウ管理プログラムに対してウィンドウのグループを指定するものであり、前記ウィンドウ管理プログラムは、前記アプリケーションプログラムからウィンドウの表示要求を受けて当該ウィンドウの表示を行う際に、当該ウィンドウのスタック順をグループ毎に一連とする制御を行う（Ｓ３０８，図４）ものであることを特徴とするウィンドウ管理装置である。

図面の簡単な説明

図１は、本発明の実施の形態１のウィンドウスタック制御方法を実現するシステムの構成例を示す図である。

図２は、本発明実施の形態１のウィンドウ情報の一例を示す図である。

図３は、本発明実施の形態１のマップ要求イベント（ＭａｐＲｅｑｕｅｓｔ）を受け取った際のウィンドウマネージャの動作例となるフロー図である。

図４は、本発明の実施の形態１の図３のステップＳ３０８のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図である。

図５は、本発明の実施の形態１の図３のステップＳ３１０のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図である。

図６は、本発明の実施の形態１の図３のステップＳ３１１のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図である。

図７は、本発明の実施の形態１のスタック最上位移動要求イベントを受け取った際のウィンドウマネージャの動作例となるフロー図である。

図８は、本発明の実施の形態１の図７のステップＳ７０５のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図である。

図９は、本発明の実施の形態１の図７のステップＳ７０８のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図である。

図１０は、本発明の実施の形態１の図７のステップＳ７１０のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図である。

図１１は、本発明の実施の形態１の図７のステップＳ７１１のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図である。

図１２は、本発明の実施の形態１のスタック最下位移動要求イベントを受け取った際のウィンドウマネージャの動作例となるフロー図である。

図１３は、本発明の実施の形態１の図１２のステップＳ１２０５のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図である。

図１４は、本発明の実施の形態１の図１２のステップＳ１２０８のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図である。

図１５は、本発明の実施の形態１の図１２のステップＳ１２１０のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図である。

図１６は、本発明の実施の形態１の図１２のステップＳ１２１１のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図である。

図１７は、本発明の実施の形態１のグループ単位スタック最上位移動要求メッセージを受け取った際のウィンドウマネージャの動作例となるフロー図である。

図１８は、本発明の実施の形態１の図１７のステップＳ１７０５のスタック変更要求によってスタックがどのよう

に変化するかを示す具体例となる模式図である。

図１９は、本発明の実施の形態１の図１７のステップＳ１７０６のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図である。

図２０は、本発明の実施の形態１のグループ単位スタック最下位移動要求メッセージを受け取った際のウィンドウマネージャの動作例となるフロー図である。

図２１は、本発明の実施の形態１の図２０のステップＳ２００４のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図である。

図２２は、本発明の実施の形態１の代表ウィンドウ使用下でのスタックの様子を示す例となる模式図である。

図２３は、本発明の実施の形態１の最上位グループに属する全ウィンドウの中のスタック最下位のウィンドウの直下に、ウィンドウ管理プログラムが生成及びマップする特定のウィンドウを位置させて使用した場合のスタックの様子を示す例となる模式図である。

図２４は、本発明の実施の形態１の最上位グループに属する全ウィンドウの中のスタック最下位のウィンドウの直下に、ウィンドウ管理プログラムが生成及びマップする特定のウィンドウを位置させて使用した場合のスクリーンに表示されている例となる模式図である。

図２５は、本発明の実施の形態２のマップ要求イベント（ＭａｐＲｅｑｕｅｓｔ）を受け取った際のウィンドウマネージャの動作例となるフロー図である。

図２６は、本発明の実施の形態２の図２５のステップＳ

２５１０のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図である。

図２７は、本発明の実施の形態２の図２５のステップＳ２５１２のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図である。

図２８は、本発明の実施の形態２の図２５のステップＳ２５１３のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図である。

図２９は、従来例におけるコンピュータ上で動作する３種類のアプリケーションが合計５枚のウィンドウを表示装置であるスクリーンに表示する際の表示例となる模式図である。

図３０は、従来例におけるウィンドウスタック順の例となる模式図である。

図３１は、従来例におけるウィンドウスタック順の例となる模式図である。

発明を実施するための最良の形態

以下、本発明の実施の形態について、図面を用いて説明する。

（実施の形態１）

本実施の形態では、ウィンドウ管理プログラムが、Ｘウィンドウシステムとウィンドウマネージャから成り、ウィンドウマネージャがルートウィンドウの子ウィンドウであるトップレベルウィンドウのスタック順を制御する場合を例にとって説明する。

図１に本発明の実施の形態１のウィンドウスタック制御方法を実現するシステムの構成例を示す。

図１において、７はコンピュータである。８はＣＰＵ（Ｃｅｎｔｒａｌ　Ｐｒｏｃｅｓｓｉｎｇ　Ｕｎｉｔ）である。９は各種プログラムや処理データ等を格納するメモリである。１０は各構成要素に所望の双方向通信を行わせるバスである。１１はＣＲＴモニタや液晶モニタといった表示装置である。１は表示装置１１のスクリーンである。１０１，１０２及び１０３はＣＰＵ８で実行されるプログラムであって、１枚以上のウィンドウをウィンドウ管理プログラム１０４を介してスクリーン１に表示しようとするアプリケーションプログラムである。１０４はＣＰＵ８で実行されるプログラムであって、アプリケーションプログラム１０１～１０３が表示しようとするウィンドウを管理してバス１０を介して表示を行うウィンドウ管理プログラムである。ウィンドウ管理プログラム１０４は、ウィンドウシステム１２とウィンドウマネージャ１０５から成る。ウィンドウシステム１２は従来のＸウィンドウシステムであり、１０５はウィンドウマネージャである。

以下、図面を用いて本実施の形態の動作を説明する。

アプリケーションプログラム１０１、１０２、１０３は、ウィンドウ管理プログラム１０４を通じて、スタック制御特性の異なる２種類のトップレベルウィンドウを表示することができる。その２種類を通常ウィンドウ、優先ウィンドウと呼ぶ事とする。両者の違いは、表示された優先ウィンドウは表示された全ての通常ウィンドウより常にスタ

ック上位に表示されることと、通常ウィンドウはグループ単位でスタック順を一連とするが優先ウィンドウについてはそのような処理を行わないことである。

＜ウィンドウ生成要求，ウィンドウ種類及びグループ設定要求，マップ要求＞

アプリケーションプログラム１０１、１０２、１０３は、ウィンドウを表示する場合、ウィンドウシステム１２に対して、ウィンドウ生成要求と、ウィンドウ種類及びグループ設定要求と、マップ要求とを発行する。この内、ウィンドウ種類及びグループ設定要求については、Ｘウィンドウシステムの場合、プロパティ機構を用いる事で、ウィンドウ毎に値を設定することができる。

各ウィンドウは、ウィンドウ生成及びマップという過程を経て初めてスクリーン１に表示されるようになる。

ウィンドウシステム１２は、ウィンドウ生成要求を受け取ると、ウィンドウの重なりの順番であるスタック順の最上位にそのウィンドウを位置させ、ウィンドウマネージャ１０５に対してウィンドウ生成通知イベント（ＣｒｅａｔｅＮｏｔｉｆｙ）を発行する。

ウィンドウマネージャ１０５は、内部データベースとして、ウィンドウ毎にウィンドウ情報を保持する。

図２にウィンドウ情報の一例を示す。

図２に示されるように、ウィンドウ情報は、ウィンドウ識別子、スタック位置確定フラグ、スタック位置情報、プロパティ獲得フラグ、ウィンドウ種類、グループ値から成る。ウィンドウ識別子は、そのウィンドウ情報がどのウィ

ンドウの情報かを示す。スタック位置確定フラグは、そのウィンドウが、ウィンドウマネージャ１０５にとって、スタック位置が暫定か確定済みなのかを示す。スタック位置情報は、そのウィンドウのスタック位置を示す。プロパティ獲得フラグは、そのウィンドウのウィンドウ種類とグループ値をウィンドウマネージャ１０５が獲得済みかどうかを示す。ウィンドウ種類は、前記プロパティ獲得フラグが獲得済みとなっている場合は、そのウィンドウが優先ウィンドウか通常ウィンドウのどちらであるかを示す。グループ値は、前記プロパティ獲得フラグが獲得済みとなっている場合は、そのウィンドウのグループ値を示す。

図１のウィンドウマネージャ１０５は、ウィンドウ生成通知イベント（ＣｒｅａｔｅＮｏｔｉｆｙ）を受け取ることで、新規にウィンドウが生成された事を知り、そのウィンドウ生成通知イベント（ＣｒｅａｔｅＮｏｔｉｆｙ）からウィンドウ識別子を獲得する。そして、内部データベースに新規にウィンドウ情報格納領域を設け、そのウィンドウ情報のウィンドウ識別子に格納する。さらにウィンドウマネージャ１０５は、同ウィンドウのスタック位置確定フラグを暫定に、スタック位置情報をスタック最上位に、プロパティ獲得フラグを未獲得に、各々初期化する。

次に、ウィンドウシステム１２は、ウィンドウ種類及びグループ設定要求を受け取り、該ウィンドウのプロパティとして記憶する。

次に、ウィンドウシステム１２は、アプリケーションプログラム１０１、１０２、１０３からマップ要求を受け取

り、内部処理は特にせずに、ウィンドウマネージャ１０５にマップ要求イベント（ＭａｐＲｅｑｕｅｓｔ）を発行する。

マップ要求イベント（ＭａｐＲｅｑｕｅｓｔ）を受け取った際のウィンドウマネージャ１０５の動作例となるフロー図を図３に示す。

図３に示されるように、ステップＳ３０１では、ウィンドウマネージャ１０５は、該時点で受け取ったマップ要求イベント（ＭａｐＲｅｑｕｅｓｔ）からマップ対象ウィンドウ（以下、単に対象ウィンドウと称す）の識別子を獲得して、ステップＳ３０２に進む。

ステップＳ３０２では、対象ウィンドウの識別子で内部データベースを検索し、対応するウィンドウ情報が存在する場合はステップＳ３０３へ進み、存在しない場合はステップＳ３１３へ進む。

ステップＳ３０３では、内部データベースで対象ウィンドウのスタック位置確定フラグが確定済みかどうかをチェックし、確定済みの場合はステップＳ３１２に進み、暫定の場合はステップＳ３０４に進む。

ステップＳ３０４では、対象ウィンドウのプロパティ獲得フラグをチェックし、獲得済みの場合はステップＳ３０６に進み、未獲得の場合はステップＳ３０５に進む。

ステップＳ３０５では、対象ウィンドウのウィンドウ種類及びグループ値をウィンドウシステム１２に問い合わせて獲得し、内部データベースに格納し、ステップＳ３０６に進む。

ステップＳ３０６では、対象ウィンドウのウィンドウ種類をチェックし、優先ウィンドウの場合はステップＳ３１２に進み、通常ウィンドウの場合はステップＳ３０７に進む。

ステップＳ３０７では、内部データベースに、対象ウィンドウと同じグループ値を持つスタック位置確定フラグが確定済みの通常ウィンドウがあるかどうかをチェックし、有る場合はステップＳ３０８進み、無い場合はステップＳ３０９に進む。

ステップＳ３０８では、対象ウィンドウのスタック位置が、同じグループ内のスタック位置確定済みの通常ウィンドウの中でスタック最上位であるウィンドウの直上となるように、ウィンドウシステム１２にスタック変更要求を発行する。

ステップＳ３０８のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図を図４に示す。

図４に示されるように、通常ウィンドウである対象ウィンドウが、対象ウィンドウと同じグループ値を持つスタック位置確定フラグが確定済みのウィンドウの内のスタック最上位のウィンドウの直上に移されているのが解る。

ステップＳ３０９では、内部データベースから、通常ウィンドウ全体を対象にスタック位置確定フラグが確定済みのウィンドウを検索し、存在する場合はステップＳ３１０に進み、存在しない場合はステップＳ３１１に進む。

ステップＳ３１０では、内部データベースにあるスタッ

ク位置確定フラグが確定済みの通常ウィンドウのうち、スタック位置が最上位のウィンドウの直上に対象ウィンドウが位置するよう、ウィンドウシステム１２に対してスタック変更要求を発行し、ステップＳ３１２に進む。

ステップＳ３１０のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図を図５に示す。

図５に示されるように、通常ウィンドウである対象ウィンドウが、スタック位置確定フラグが確定済みの通常ウィンドウ全体の内のスタック最上位のウィンドウの直上に移されているのが解る。

ステップＳ３１１では、スタック位置の最下位に対象ウィンドウが位置するよう、ウィンドウシステム１２にスタック変更要求を発行し、ステップＳ３１２に進む。

ステップＳ３１１のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図を図６に示す。

図６に示されるように、通常ウィンドウである対象ウィンドウが、スタック最下位に移されているのが解る。

ステップＳ３１２では、ウィンドウシステム１２に対して対象ウィンドウのマップ要求を発行し、内部データベースの対象ウィンドウのウィンドウ情報の内、スタック位置確定フラグが暫定を示している場合は確定済みに更新し、また、内部データベース全体に対して必要に応じてスタック位置情報を更新した上で、ステップＳ３１３に進む。

ステップＳ３１３では、処理を終了する。

ウィンドウシステム１２は、図３のステップＳ３０８、ステップＳ３１０、ステップＳ３１１に起因するスタック変更要求をウィンドウマネージャ１０５から受け取ると、要求に従ってウィンドウのスタック順を変更する。また、図３のステップＳ３１２に起因するマップ要求をウィンドウマネージャ１０５から受け取ると、要求に従ってウィンドウのマップする。マップされたウィンドウは、ウィンドウの重なり具合にもよるが、他のウィンドウの下に隠れていなければ、スクリーン１に表示されるようになる。

図３の処理によって、アプリケーション１０１、１０２、１０３が生成及びウィンドウ種類及びグループ設定した通常ウィンドウは、表示までに必ず、スタック位置情報が確定済みとなり、また、スタック位置がグループ毎に纏められ、また、優先ウィンドウのスタック上位に表示される事もない。また、優先ウィンドウは、スタック位置が変更されることなく表示される。

＜グループ内スタック最上位移動要求＞

図１において、アプリケーションプログラム１０１，１０２，１０３は、生成及びウィンドウ種類及びグループ設定した通常ウィンドウに対して、スタック変更要求として、グループ内スタック最上位移動要求をウィンドウシステム１２に対して発行する事ができる。Ｘウィンドウシステムの場合、例えば、グループ内スタック最上位移動要求にＸＲａｉｓｅＷｉｎｄｏｗ（）関数を割り当てれば良い。また、優先ウィンドウに対して、スタック変更要求として、優先ウィンドウスタック最上位移動要求をウィンドウシ

ステム１２に対して発行する事ができる。例えば、グループ内スタック最上位移動要求と同じＸＲａｉｓｅＷｉｎｄｏｗ（）関数を割り当てれば良い。

アプリケーションプログラムがＸＲａｉｓｅＷｉｎｄｏｗ（）を実施した場合、ウィンドウシステム１２はスタック変更をせず、ウィンドウマネージャ１０５に対してスタック最上位移動要求イベント（ＣｏｎｆｉｇｕｒｅＲｅｑｕｅｓｔ）を発行する。

スタック最上位移動要求イベントを受け取った際のウィンドウマネージャ１０５の動作例となるフロー図を図７に示す。

図７に示されるように、ステップＳ７０１では、該時点で受け取ったスタック最上位移動要求イベント（ＣｏｎｆｉｇｕｒｅＲｅｑｕｅｓｔ）から対象ウィンドウの識別子を獲得して、ステップＳ７０２に進む。

ステップＳ７０２では、対象ウィンドウの識別子で内部データベースを検索し、対応するデータが存在する場合はステップＳ７０３へ進み、存在しない場合はステップＳ７１２へ進む。

ステップＳ７０３では、内部データベースにあるプロパティ獲得フラグが未獲得である全ウィンドウについて、ウィンドウシステム１２に対してウィンドウ種類及びグループ値を問い合わせ、獲得できたウィンドウについては、そのウィンドウ種類及びグループ値と、プロパティ獲得済みである事を内部データベースに記憶し、ステップＳ７０４に進む。

ステップＳ７０４では、対象ウィンドウのウィンドウ種類をチェックし、優先ウィンドウであればステップＳ７０５に進み、通常ウィンドウであればステップＳ７０６に進む。

ステップＳ７０５では、内部データベースからウィンドウ種類が優先ウィンドウであるものの内、スタック最上位のウィンドウを検索する。そして、対象ウィンドウが、そのスタック最上位のウィンドウの直上となるよう、ウィンドウシステム１２にスタック変更要求を発行し、次に、内部データベース全体に対してスタック位置情報を更新した上で、ステップＳ７１２に進む。なお、検索の結果、スタック最上位のウィンドウが対象ウィンドウ自身であった場合は、何もせずステップＳ７１２に進む。

ステップＳ７０５のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図を図８に示す。

図８に示されるように、対象ウィンドウのみが、優先ウィンドウの中のスタック最上位のウィンドウの直上に移されているのが解る。

図７において、ステップＳ７０６では、対象ウィンドウと同じグループに属する通常ウィンドウを内部データベースから検索してリストアップし、ステップＳ７０７に進む。

ステップＳ７０７では、前記リストアップした中にスタック位置確定フラグが確定済みであるウィンドウが存在するならばステップＳ７０８に進み、存在しないならステッ

プＳ７０９に進む。

ステップＳ７０８では、リストアップした中でスタック位置確定フラグが暫定であるウィンドウがあれば、それら全てを、それらの中でみたスタック順はそのままで、リストアップした中でスタック位置確定フラグが確定済みであるウィンドウの内の最上位のウィンドウの直上に位置させ、更に対象ウィンドウをリストアップしたウィンドウの中で最上位となるよう、ウィンドウシステム１２にスタック変更要求を発行し、次に、リストアップした中でスタック位置確定フラグが暫定であるウィンドウがあれば確定済みに更新し、また、内部データベース全体に対してスタック位置情報を更新した上で、ステップＳ７１０に進む。

ステップＳ７０８のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図を図９に示す。

図９に示されるように、対象ウィンドウと同じグループ値を持つスタック位置確定フラグが暫定のウィンドウが、同じグループ値を持つスタック位置確定フラグが確定済みのウィンドウの直上に移され、同一グループ値を持つウィンドウ全体の中で対象ウィンドウのスタック最上位移動がなされているのが解る。

図７において、ステップＳ７０９では、内部データベースにスタック位置確定フラグが確定済みのウィンドウが存在するならばステップＳ７１０に進み、存在しないならばステップＳ７１１に進む。

ステップＳ７１０では、ステップＳ７０６でリストアッ

プした中でスタック位置確定フラグが暫定であるウィンドウ全てを、それらの中でみたスタック順はそのままで、内部データベースの中でスタック位置確定フラグが確定済みであるスタック最上位の通常ウィンドウの直上に位置させ、更に対象ウィンドウが前記リストアップしたウィンドウの中で最上位となるよう、ウィンドウシステム１２にスタック変更要求を発行する。そして、リストアップしたウィンドウのスタック位置確定フラグを確定済みに更新し、さらに、内部データベース全体に対してスタック位置情報を更新した上で、ステップＳ７１２に進む。

ステップＳ７１０のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図を図１０に示す。

図１０に示されるように、対象ウィンドウと同じグループ値を持つスタック位置確定フラグが暫定の通常ウィンドウが、スタック位置確定フラグが確定済みの通常ウィンドウ全体のスタック最上位ウィンドウの直上に移され、同一グループ値を持つウィンドウ全体の中で対象ウィンドウのスタック最上位移動がなされているのが解る。

図７において、ステップＳ７１１では、ステップＳ７０６でリストアップした中でスタック位置確定フラグが暫定であるウィンドウ全てを、それらの中でみたスタック順はそのままで、全体スタックの最下位とし、更に対象ウィンドウをリストアップしたウィンドウの中で最上位となるよう、ウィンドウシステム１２にスタック変更要求を発行し、次に、リストアップしたウィンドウのスタック位置確定

フラグを確定済みに更新し、また、内部データベース全体に対してスタック位置情報を更新した上で、ステップＳ７１２に進む。

ステップＳ７１１のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図を図１１に示す。

図１１に示されるように、対象ウィンドウと同じグループ値を持つスタック位置確定フラグが暫定の通常ウィンドウが、スタック最下位に移され、同一グループ値を持つ通常ウィンドウ全体の中で対象ウィンドウのスタック最上位移動がなされているのが解る。

図７において、ステップＳ７１２では、処理を終了する。

＜グループ内スタック最下位移動要求＞

図１において、アプリケーションプログラム１０１，１０２，１０３は、生成及びウィンドウ種類及びグループ設定したウィンドウに対して、グループ内スタック最下位移動要求をウィンドウシステム１２に対して発行する事ができる。Ｘウィンドウシステムの場合、例えば、グループ内スタック最下位移動要求にＸＬｏｗｅｒＷｉｎｄｏｗ（）関数を割り当てれば良い。また、優先ウィンドウに対して、スタック変更要求として、優先ウィンドウスタック最下位移動要求をウィンドウシステム１２に対して発行する事ができる。例えば、グループ内スタック最下位移動要求と同じＸＬｏｗｅｒＷｉｎｄｏｗ（）関数を割り当てれば良い。

アプリケーションプログラムがＸＬｏｗｅｒＷｉｎｄｏｗ（）を実施した場合、ウィンドウシステム１２はスタック変更をせず、ウィンドウマネージャ１０５に対してスタック最下位移動要求イベント（ＣｏｎｆｉｇｕｒｅＲｅｑｕｅｓｔ）を発行する。

スタック最下位移動要求イベントを受け取った際のウィンドウマネージャ１０５の動作例となるフロー図を図１２に示す。

図１２に示されるように、ステップＳ１２０１では、該時点で受け取ったスタック最下位移動要求イベント（ＣｏｎｆｉｇｕｒｅＲｅｑｕｅｓｔ）から対象ウィンドウの識別子を獲得して、ステップＳ１２０２に進む。

ステップＳ１２０２では、対象ウィンドウの識別子で内部データベースを検索し、対応するデータが存在する場合はステップＳ１２０３へ進み、存在しない場合はステップＳ１２１２へ進む。

ステップＳ１２０３では、内部データベースにあるプロパティ獲得フラグが未獲得である全ウィンドウについて、ウィンドウシステム１２に対してウィンドウ種類及びグループ値を問い合わせ、獲得できたウィンドウについては、そのウィンドウ種類及びグループ値と、プロパティ獲得済みである事を内部データベースに記憶し、ステップＳ１２０４に進む。

ステップＳ１２０４では、対象ウィンドウのウィンドウ種類をチェックし、優先ウィンドウの場合はステップＳ１２０５へ進み、通常ウィンドウの場合はステップＳ１２０

６へ進む。

ステップＳ１２０５では、内部データベースからウィンドウ種類が優先ウィンドウであるものの内、スタック最下位のウィンドウを検索する。そして、対象ウィンドウが、そのスタック最下位のウィンドウの直下となるよう、ウィンドウシステム１２にスタック変更要求を発行し、次に、内部データベース全体に対してスタック位置情報を更新した上で、ステップＳ１２１２に進む。なお、検索の結果、スタック最下位のウィンドウが対象ウィンドウ自身であった場合は、何もせずステップＳ１２１２へ進む。

ステップＳ１２０５のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図を図１３に示す。

図１３に示されるように、対象ウィンドウのみが、優先ウィンドウの中のスタック最下位のウィンドウの直下に移されているのが解る。

図１２において、ステップＳ１２０６では、対象ウィンドウと同じグループに属する通常ウィンドウを内部データベースから検索してリストアップし、ステップＳ１２０７に進む。

ステップＳ１２０７では、前記リストアップした中にスタック位置確定フラグが確定済みであるウィンドウが存在するならばステップＳ１２０８に進み、存在しないならステップＳ１２０９に進む。

ステップＳ１２０８では、リストアップした中でスタック位置確定フラグが暫定であるウィンドウがあれば、それ

ら全てを、それらの中でみたスタック順はそのままで、リストアップした中でスタック位置確定フラグが確定済みであるウィンドウの内の最上位のウィンドウの直上に位置させ、更に対象ウィンドウをリストアップしたウィンドウの中で最下位となるよう、ウィンドウシステム１２にスタック変更要求を発行し、次に、リストアップした中でスタック位置確定フラグが暫定であるウィンドウがあれば確定済みに更新し、また、内部データベース全体に対してスタック位置情報を更新した上で、ステップＳ１２１２に進む。

ステップＳ１２０８のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図を図１４に示す。

図１４に示されるように、対象ウィンドウと同じグループ値を持つスタック位置確定フラグが暫定の通常ウィンドウが、同じグループ値を持つスタック位置確定フラグが確定済みの通常ウィンドウの直上に移され、同一グループ値を持つ通常ウィンドウ全体の中で対象ウィンドウのスタック最下位移動がなされているのが解る。

図１２において、ステップＳ１２０９では、内部データベースにスタック位置確定フラグが確定済みの通常ウィンドウが存在するならばステップＳ１２１０に進み、存在しないならばステップＳ１２１１に進む。

ステップＳ１２１０では、ステップＳ１２０６でリストアップした中でスタック位置確定フラグが暫定であるウィンドウ全てを、それらの中でみたスタック順はそのままで、内部データベースの中でスタック位置確定フラグが確定

済みであるスタック最上位の通常ウィンドウの直上に位置させ、更に対象ウィンドウを前記リストアップしたウィンドウの中で最下位となるよう、ウィンドウシステム１２にスタック変更要求を発行し、次に、リストアップしたウィンドウのスタック位置確定フラグを確定済みに更新し、また、内部データベース全体に対してスタック位置情報を更新した上で、ステップＳ１２１２に進む。

ステップＳ１２１０のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図を図１５に示す。

図１５に示されるように、対象ウィンドウと同じグループ値を持つスタック位置確定フラグが暫定の通常ウィンドウが、スタック位置確定フラグが確定済みの通常ウィンドウ全体のスタック最上位ウィンドウの直上に移され、同一グループ値を持つウィンドウ全体の中で対象ウィンドウのスタック最下位移動がなされているのが解る。

図１２において、ステップＳ１２１１では、ステップＳ１２０６でリストアップした中でスタック位置確定フラグが暫定であるウィンドウ全てを、それらの中でみたスタック順はそのままで、全体スタックの最下位とし、更に対象ウィンドウをリストアップしたウィンドウの中で最下位となるよう、ウィンドウシステム１２にスタック変更要求を発行し、次に、リストアップしたウィンドウのスタック位置確定フラグを確定済みに更新し、また、内部データベース全体に対してスタック位置情報を更新した上で、ステップＳ１２１２に進む。

ステップＳ１２１１のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図を図１６に示す。

図１６に示されるように、対象ウィンドウと同じグループ値を持つスタック位置確定フラグが暫定の通常ウィンドウが、スタック最下位に移され、同一グループ値を持つ通常ウィンドウ全体の中で対象ウィンドウのスタック最下位移動がなされているのが解る。

図１２において、ステップＳ１２１２では、処理を終了する。

＜グループ単位スタック最上位移動要求＞

図１において、アプリケーションプログラム１０１，１０２，１０３は、生成及びウィンドウ種類及びグループ設定した通常ウィンドウに対して、グループ単位スタック最上位移動要求をウィンドウシステム１２に対して発行する事ができる。Ｘウィンドウシステムの場合、例えば、クライアントメッセージ機構を利用する事で容易に実現できる。今後、クライアントメッセージでウィンドウシステム１２に対して発行するグループ単位スタック最上位移動要求のメッセージの事をグループ単位スタック最上位移動要求メッセージと呼ぶ事とする。そして、このグループ単位スタック最上位移動要求メッセージには、移動したいグループ（対象グループ）のグループ値が含まれているものとする。

アプリケーションプログラム１０１，１０２，１０３がグループ単位スタック最上位移動要求メッセージ（Ｃｌｉ

ｅｎｔＭｅｓｓａｇｅ）を発行した場合、このメッセージはウィンドウシステム１２経由でウィンドウマネージャ１０５に転送される。

グループ単位スタック最上位移動要求メッセージを受け取った際のウィンドウマネージャ１０５の動作例となるフローを図１７に示す。

図１７に示されるように、ステップＳ１７０１では、該時点で受け取ったグループ単位スタック最上位移動要求メッセージ（ＣｌｉｅｎｔＭｅｓｓａｇｅ）から対象グループ値を獲得して、ステップＳ１７０２に進む。

ステップＳ１７０２では、内部データベースにあるプロパティ獲得フラグが未獲得である全ウィンドウについて、ウィンドウシステム１２に対してウィンドウ種類及びグループ値を問い合わせ、獲得できたウィンドウについては、そのウィンドウ種類及びグループ値と、プロパティ獲得済みである事を内部データベースに記憶し、ステップＳ１７０３に進む。

ステップＳ１７０３では、対象グループ値を持つ通常ウィンドウ情報を内部データベースから検索してリストアップし、ステップＳ１７０４に進む。

ステップＳ１７０４では、内部データベースにスタック位置確定情報が確定済み且つグループ値情報が対象グループ値とは異なる通常ウィンドウが存在すればステップＳ１７０５に進み、存在しなければステップＳ１７０６に進む。

ステップＳ１７０５では、リストアップしたウィンドウ

が、それらの中でのスタック順はそのままで、内部データベースの中でスタック位置確定フラグが確定済み且つグループ値情報が対象グループ値とは異なる通常ウィンドウの内でスタック最上位のウィンドウの直上となるよう、ウィンドウシステム１２にスタック変更要求を発行し、次に、リストアップした中でスタック位置確定フラグが暫定であるウィンドウがあれば確定済みに更新し、また、内部データベース全体に対してスタック位置情報を更新した上で、ステップＳ１７０７に進む。

ステップＳ１７０５のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図を図１８に示す。

図１８に示されるように、対象グループ値を持つ通常ウィンドウが、スタック位置確定フラグが確定済み且つグループ値情報が対象グループ値とは異なる通常ウィンドウ全体のスタック最上位ウィンドウの直上に移されているのが解る。

図１７におけるステップＳ１７０６では、リストアップしたウィンドウが、それらの中でのスタック順はそのままで、全体スタックの最下位となるよう、ウィンドウシステム１２にスタック変更要求を発行し、次に、リストアップした中でスタック位置確定フラグが暫定であるウィンドウがあれば確定済みに更新し、また、内部データベース全体に対してスタック位置情報を更新した上で、ステップＳ１７０７に進む。

ステップＳ１７０６のスタック変更要求によってスタッ

クがどのように変化するかを示す具体例となる模式図を図１９に示す。

図１９に示されるように、対象グループ値を持つ通常ウィンドウが、全体スタックの最下位に移されているのが解る。

図１７において、ステップＳ１７０７では、処理を終了する。

＜グループ単位スタック最下位移動要求＞

図１において、アプリケーションプログラム１０１，１０２，１０３は、生成及びウィンドウ種類及びグループ設定した通常ウィンドウに対して、グループ単位スタック最下位移動要求をウィンドウシステム１２に対して発行する事ができる。Ｘウィンドウシステムの場合、例えば、クライアントメッセージ機構を利用する事で容易に実現できる。今後、クライアントメッセージでウィンドウシステム１２に対して発行するグループ単位スタック最下位移動要求のメッセージの事をグループ単位スタック最下位移動要求メッセージと呼ぶ事とする。そして、このグループ単位スタック最下位移動要求メッセージには、移動したいグループ（対象グループ）のグループ値が含まれているものとする。

アプリケーションプログラム１０１，１０２，１０３がグループ単位スタック最下位移動要求メッセージ（ＣｌｉｅｎｔＭｅｓｓａｇｅ）を発行した場合、このメッセージはウィンドウシステム１２経由でウィンドウマネージャ１０５に転送される。

グループ単位スタック最下位移動要求メッセージを受け取った際のウィンドウマネージャ１０５の動作例となるフローを図２０に示す。

図２０に示されるように、ステップＳ２００１では、該時点で受け取ったグループ単位スタック最下位移動要求メッセージ（ＣｌｉｅｎｔＭｅｓｓａｇｅ）から対象グループ値を獲得して、ステップＳ２００２に進む。

ステップＳ２００２では、内部データベースにあるプロパティ獲得フラグが未獲得である全ウィンドウについて、ウィンドウシステム１２に対してウィンドウ種類及びグループ値を問い合わせ、獲得できたウィンドウについては、そのウィンドウ種類及びグループ値と、プロパティ獲得済みである事を内部データベースに記憶し、ステップＳ２００３に進む。

ステップＳ２００３では、対象グループ値を持つ通常ウィンドウを内部データベースから検索してリストアップし、ステップＳ２００４に進む。

ステップＳ２００４では、リストアップしたウィンドウが、それらの中でのスタック順はそのままで、全体スタックの最下位となるよう、ウィンドウシステム１２にスタック変更要求を発行し、次に、リストアップした中でスタック位置確定フラグが暫定であるウィンドウがあれば確定済みに更新し、また、内部データベース全体に対してスタック位置情報を更新した上で、ステップＳ２００５に進む。

ステップＳ２００４のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図を図

２１に示す。

図２１に示されるように、対象グループ値を持つ通常ウィンドウが、全体スタックの最下位に移されているのが解る。

図２０におけるステップＳ２００５では、処理を終了する。

＜破棄要求＞

図１において、アプリケーションプログラム１０１，１０２，１０３は、生成したウィンドウに対して破棄要求を発行する。

ウィンドウシステム１２は、破棄要求を受け取ると、該当するウィンドウを管理から外して削除する。そして、ウィンドウマネージャ１０５にウィンドウ破棄通知イベント（ＤｅｓｔｒｏｙＮｏｔｉｆｙ）を発行する。

ウィンドウマネージャ１０５は、ウィンドウ破棄通知イベント（ＤｅｓｔｒｏｙＮｏｔｉｆｙ）を受け取ると、そこからウィンドウ識別子を抽出して、該当するウィンドウ情報を内部データベースから検索し削除した後、内部データベース全体に対して必要に応じてスタック位置を更新し、処理を終了する。

以上のことにより、各通常ウィンドウはそのウィンドウの表示までに、ウィンドウのスタック順がグループ毎に纏められるので、最上位のグループの各ウィンドウは、他のグループのウィンドウの下に重ねられて表示される事が無くなる。また、ウィンドウ管理プログラムが、スタック変更要求を受け取る事を切っ掛けにして、スタック変更対象

となるグループに属する通常ウィンドウ全てを、その中でのスタック順は保持したままで纏めた上でスタック変更を実施するので、アプリケーションプログラムは、自グループのウィンドウのスタック順を把握し易くなる。また、ウィンドウ管理プログラムにグループ内のスタック最上位移動とグループ内のスタック最下位移動の機能を設ける事で、グループ内で通常ウィンドウのスタック順を操作できるので、各アプリケーションプログラムは、他のアプリケーションプログラムの通常ウィンドウ表示を気にする事無く、スタック変更を行う事ができる。また、ウィンドウ管理プログラムにグループ単位のスタック最上位移動と最下位移動の機能を設ける事で、例えば、グループをアプリケーションプログラム単位で振り分けた場合だと、そのアプリケーションプログラムに属するウィンドウのメンバーを知らなくても、スタック最前面（最上位）にそのアプリケーションプログラムに属する全通常ウィンドウを移動させる事が可能となる。また、優先ウィンドウを設ける事で、グループ単位のスタック順に関係無く、特定のウィンドウを最優先で表示することのできるサービスを提供することができる。

なお、本実施の形態では、ウィンドウマネージャにおけるウィンドウ種類及びグループ値取得のタイミングをマップ要求イベント（ＭａｐＲｅｑｕｅｓｔ）受信時及びスタック変更要求受信時としたが、これに限定されるものではなく、アプリケーションプログラムがウィンドウ種類及びグループ設定をしたタイミングで取得して内部データベー

スに記憶しておいても良く、その方がむしろ処理効率が良い。更に、このタイミングで、スタック順をグループ毎に纏めても構わない。また、ウィンドウ種類及びグループ値の変更を認めても良いし、認めなくても良い。認める場合は、ウィンドウ種類及びグループ設定のタイミングで、ウィンドウ種類が優先ウィンドウから通常ウィンドウに変更された場合は、スタック位置確定フラグを暫定に変更する。逆にウィンドウ種類が通常ウィンドウから優先ウィンドウに変更された場合は、該時点のスタック最上位に該ウィンドウを位置させるようにスタック変更を行う。また、ウィンドウ種類に変更が無く、グループ値が変更となった場合は、スタック位置確定フラグを暫定に変更する。そして、内部データベースを更新すれば良い。また、認めない場合は、ウィンドウ種類及びグループ設定のタイミングで、プロパティ獲得フラグをチェックして、獲得済みであれば、該設定を無視すれば良い。

また、ウィンドウ種類として優先ウィンドウと通常ウィンドウの２種類を制御する例を示したが、通常ウィンドウの１種類だけであっても容易に実施できる事は言うまでも無い。

また、ウィンドウマネージャがルートウィンドウの子ウィンドウであるトップレベルウィンドウのスタック順を制御する場合を例に示したが、トップレベルウィンドウの子孫ウィンドウに対しても実施する事ができる事は言うまでも無い。

また、ウィンドウ管理プログラムをＸウィンドウシステ

ムとウィンドウマネージャとして、アプリケーションプログラムとのインターフェースにウィンドウ生成要求、マップ要求を用いたが、これに限定されるものではなく、表示要求に相当するインターフェースさえあれば良く、その場合、ウィンドウ管理プログラムは、ウィンドウ表示要求受信時に、本実施の形態で示したウィンドウ生成要求受信時の処理とマップ要求受信時の処理を纏めて行えば良い。

また、ウィンドウのスタック順をグループ毎に一連とする際、単にウィンドウのスタック順を変更するだけであったが、例えば、ウィンドウ管理プログラムが、グループ毎に代表ウィンドウを１枚生成し、ウィンドウのスタック順をグループ毎に一連とする際に、前記ウィンドウを前記代表ウィンドウの子ウィンドウとするようにして制御することも容易である。そうすることによって、グループ単位のスタック移動やグループ単位のウィンドウの表示／非表示の切り替えを代表ウィンドウに対する制御で実施できる分、グループ内のウィンドウが多いほど、処理負荷を軽減することができる。代表ウィンドウ使用下でのスタックの様子を示す例となる模式図を図２２に示す。

また、最上位グループよりスタック下位のウィンドウ表示について、ウィンドウ管理プログラムでは非表示とするような制御はしていないが、非表示とする制御を行う事も容易に実現できる。その場合、ウィンドウ情報に表示／非表示を示すマップフラグを追加して、このマップフラグによりアプリケーションプログラムの表示要求状態を記憶しておき、該ウィンドウの所属するグループが通常ウィンド

ウの中でスタック最上位となった時、前記マップフラグをチェックし、表示となっていればマップ処理を行い、非表示となっていればアンマップのままとする。逆に、該ウィンドウの所属するグループが通常ウィンドウの中でスタック最上位でなくなった際には、前記マップフラグが表示となっていればアンマップ処理を行い、非表示となっていれば何もしない。これにより、通常ウィンドウの中の最上位グループよりスタック下位のグループが何を表示しようとしても表示装置に表示されないので、最上位グループのウィンドウで覆われない部分にスタック下位のグループのウィンドウが中途半端に映り込む心配が無い。

また、本実施形態では、通常ウィンドウの最上位グループに属する全ウィンドウの中のスタック最下位のウィンドウの直下に、ウィンドウ管理プログラムが生成及びマップする特定のウィンドウを位置させることは特にしなかったが、位置させることも容易に実現できる。これにより、例えば、前記特定のウィンドウを表示装置のスクリーン一杯に表示しておけば、前記特定のウィンドウのスタック下位のグループが何を表示しようとしても，表示装置に表示されないので、最上位グループのウィンドウで覆われない部分にスタック下位のグループのウィンドウが中途半端に映り込む心配が無い。前記特定のウィンドウを使用した場合のスタックの様子を示す例となる模式図を図２３に示し、また、スクリーンに表示されている例となる模式図を図２４に示す。

また、アプリケーションプログラムが３つの場合を例に

示したが、これに限定されるものではなく、１つ以上であれば良い。

また、各アプリケーションプログラムに１つのグループを割り当てた場合を例に示したが、これに限定されるものではなく、１つのアプリケーションに複数のグループを割り当てても良く、また、複数のアプリケーションに１つのグループを割り当てても構わない。

また、ウィンドウ管理プログラムをＸウィンドウシステムとウィンドウマネージャとで構成し、両者間でイベントや各種要求を送受信し合う例を示したが、これに限定されるものではなく、Ｘウィンドウシステムである必要は無いし、また、２つに分けてイベントや各種要求を送受信する構成である必要もなく、一体化して実施する場合は、イベントやメッセージのインターフェースを関数ベースに変える事で本実施の形態を容易に応用できる。

また、通常ウィンドウの最上位グループに属する全ウィンドウの中のスタック最上位のウィンドウの直上に、ウィンドウ管理プログラムが生成する特定のウィンドウを位置させることは特にしていないが、位置させることも容易に実現できる。これにより、最上位グループを別のグループに入れ替える際、Ｘウィンドウシステムの場合、ＸＲｅｓｔａｃｋＷｉｎｄｏｗｓ（）関数を使用する事となるが、その特定のウィンドウをウィンドウ群のスタック移動先を示すために使用できる。もし、最上位グループのウィンドウよりスタック上位にウィンドウが存在しなかった場合、存在する全てのウィンドウの並び順を決め、前記ＸＲｅｓ

ｔａｃｋＷｉｎｄｏｗｓ（）の引数に代入せねばならないが、前記特定のウィンドウがあれば、前記特定のウィンドウと前記ウィンドウ群を引数に代入するだけで済む。以下、その理由を簡単に説明する。ＸＲｅｓｔａｃｋＷｉｎｄｏｗｓ（）関数でスタック順を変更するには２通りの方法がある。１つ目の方法は、引数として、既存のウィンドウの全てをスタック順に並べる全体指定による方法である。２つ目の方法は、引数として、スタック順を変えない１つのウィンドウと、その下位のウィンドウの内のスタック順を変える部分までを指定する部分指定による方法である。例えば、スタック順上位から、Ａ，Ｂ，Ｃ，Ｄ，Ｅの５つのウィンドウが存在していて、このスタック順をＡ，Ｃ，Ｂ，Ｄ，Ｅに変更する場合、全体指定の引数としては（Ａ，Ｃ，Ｂ，Ｄ，Ｅ）を、部分指定の引数として（Ａ，Ｃ，Ｂ）を代入すればよい。このように、部分指定は、全体指定に比べて、引数に指定すべきウィンドウの数が少なくて済むというメリットがある。上記のように、通常ウィンドウの最上位グループに属する全ウィンドウの中のスタック最上位のウィンドウの直上に、ウィンドウ管理プログラムが生成する特定のウィンドウを位置させることにより、その特定のウィンドウを、部分指定の引数における「スタック順を変えない１つのウィンドウ」として利用することができるため、最上位グループを別のグループに入れ替える際にも部分指定によるスタック順の変更が可能となる。なお、部分指定の他のメリットとして、ＸＲｅｓｔａｃｋＷｉｎｄｏｗｓ（）関数の実行が成功し易くなることがある

。なぜなら、ウィンドウシステムとウィンドウマネージャとは非同期に動作しており、アプリケーションプログラムからの要求に基づくウィンドウ生成・破棄は、マップ（ウィンドウ表示）とは異なり、アプリケーションプログラムとウィンドウシステムとの間で勝手に実施されるため、ウィンドウマネージャが管理している、ウィンドウの有無に関する情報は、最新の情報とは異なっている場合があるからである。この場合、全体指定では全ウィンドウのスタック順を指定する必要があるため、スタック順の変更に失敗してしまうのである。

また、Ｘウィンドウシステムを用いる場合、ウィンドウマネージャは、スタックを操作するためにＸＲｅｓｔａｃｋＷｉｎｄｏｗｓ（）関数を使用する。ＸＲｅｓｔａｃｋＷｉｎｄｏｗｓ（）関数の引数には、ウィンドウ群とそのウィンドウ群の移動先を示すウィンドウを指定するが、それらの何れかが破棄されていた場合、スタック操作が意図した通りに実施されない事がある。そのための対策として、例えば、ウィンドウマネージャは、ウィンドウ破棄通知イベント（ＤｅｓｔｒｏｙＮｏｔｉｆｙ）を受け取った際に、該当するウィンドウ情報を内部データベースから検索し削除し、内部データベース全体に対して必要に応じてスタック位置を更新した後、次に、ウィンドウシステムの該時点でのスタックツリーを問い合わせて獲得し、ウィンドウマネージャが認識していない新規のウィンドウは対象から外し、ウィンドウマネージャのスタックツリーとの整合性をチェックし、ウィンドウマネージャの認識と異なって

いる場合は、ウィンドウマネージャのスタックツリーに合わせ込むスタック変更要求を発行し、認識が合うまで、ウィンドウシステムの該時点でのスタックツリー問い合わせ、整合性チェック、スタック変更要求の発行を繰り返すようにしても良い。これにより、ウィンドウシステムがスタック変更処理に失敗しても、ウィンドウシステム間とのスタック認識の整合性を確保でき、処理が破綻する事がなくなる。

また、ウィンドウマネージャが、ＸＲｅｓｔａｃｋＷｉｎｄｏｗｓ（）を実行した際にフラグを立て、ウィンドウ破棄通知イベント（ＤｅｓｔｒｏｙＮｏｔｉｆｙ）を受け取った際に、前記フラグが立っている場合にのみ、前記フラグを下げると共にスタックツリーの整合性のための処理を行うようにしても良い。これにより、スタックツリーの整合性のための処理の実行頻度を軽減できる。

また、コンピュータ７は、家庭用コンピュータや携帯電話、ＰＤＡ（Ｐｅｒｓｏｎａｌ　Ｄｉｇｉｔａｌ　Ａｓｓｉｓｔａｎｃｅ）、セットトップボックス、デジタルスチルカメラ、カメラ一体型ＶＴＲ等、プログラムが動作するためのＣＰＵとメモリを実装して表示装置に複数のウィンドウを表示しようとする装置であれば何でも良い事は言うまでもない。

また、表示装置１１をコンピュータ７の一部としたが、物理上で一体化している事を指すものではなく、コンピュータ７と有線／無線を問わず、何らかの形で接続されていさえすれば良い。

また、本実施の形態のウィンドウスタック制御方法を実現するためのウィンドウ管理プログラムは、ＲＯＭやディスク等に記憶してコンピュータに供給する事ができるほか、ネットワークを介してコンピュータに供給する事が可能である事は言うまでもない。

（実施の形態２）

実施の形態２のウィンドウスタック制御方法は、実施の形態１を変形したものであり、実施の形態１と実施の形態２の違いは、マップ要求イベント（ＭａｐＲｅｑｕｅｓｔ）を受け取った際のウィンドウマネージャ１０５の動作と、アプリケーションプログラムがウィンドウ種類及びグループ値を設定するタイミングでウィンドウマネージャ１０５はスタック順をグループ毎に纏めてはいけないことだけである。よって、本実施例では、ウィンドウマネージャ１０５のマップ要求イベント（ＭａｐＲｅｑｕｅｓｔ）を受け取った際のウィンドウマネージャ１０５の動作についてのみ説明する。

マップ要求イベント（ＭａｐＲｅｑｕｅｓｔ）を受け取った際のウィンドウマネージャ１０５の動作例となるフロー図を図２５に示す。

図２５に示されるように、ステップＳ２５０１では、該時点で受け取ったマップ要求イベント（ＭａｐＲｅｑｕｅｓｔ）からマップ対象ウィンドウの識別子を獲得して、ステップＳ２５０２に進む。

ステップＳ２５０２では、マップ対象ウィンドウの識別子で内部データベースを検索し、対応するウィンドウ情報

が存在する場合はステップＳ２５０３へ進み、存在しない場合はステップＳ２５１５へ進む。

ステップＳ２５０３では、内部データベースで対象ウィンドウのスタック位置確定フラグが確定済みかどうかをチェックし、確定済みの場合はステップＳ２５１４に進み、暫定の場合はステップＳ２５０４に進む。

ステップＳ２５０４では、対象ウィンドウのプロパティ獲得フラグをチェックし、獲得済みの場合はステップＳ２５０６に進み、未獲得の場合はステップＳ２５０５に進む。

ステップＳ２５０５では、対象ウィンドウのウィンドウ種類及びグループ値をウィンドウシステムに問い合わせて獲得し、内部データベースに格納し、ステップＳ２５０６に進む。

ステップＳ２５０６では、対象ウィンドウのウィンドウ種類をチェックし、優先ウィンドウの場合は、ステップＳ２５１４に進み、通常ウィンドウの場合は、ステップＳ２５０７に進む。

ステップＳ２５０７では、内部データベースにあるプロパティ獲得フラグが未獲得である全ウィンドウについて、ウィンドウシステム１２に対してウィンドウ種類及びグループ値を問い合わせ、獲得できたウィンドウについては、そのウィンドウ種類及びグループ値と、プロパティ獲得済みである事を内部データベースに記憶し、ステップＳ２５０８に進む。

ステップＳ２５０８では、対象ウィンドウと同じグルー

プに属する通常ウィンドウを内部データベースから検索してリストアップし、ステップＳ２５０９に進む。

ステップＳ２５０９では、前記リストアップした中にスタック位置確定フラグが確定済みであるウィンドウが存在するならばステップＳ２５１０に進み、存在しないならステップＳ２５１１に進む。

ステップＳ２５１０では、リストアップした中でスタック位置確定フラグが暫定であるウィンドウ全てを、それらの中でみたスタック順はそのままで、リストアップした中でスタック位置確定フラグが確定済みであるウィンドウの内の最上位のウィンドウの直上となるよう、ウィンドウシステム１２にスタック変更要求を発行し、ステップＳ２５１４に進む。

ステップＳ２５１０のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図を図２６に示す。

図２６に示されるように、マップ対象ウィンドウ及びマップ対象ウィンドウと同じグループ値を持つスタック位置確定フラグが暫定の通常ウィンドウが、前記グループ値を持つスタック位置確定フラグが確定済みの通常ウィンドウの直上に移されているのが解る。

図２５において、ステップＳ２５１１では、内部データベースにスタック位置確定フラグが確定済みの通常ウィンドウが存在するならばステップＳ２５１２に進み、存在しないならばステップＳ２５１３に進む。

ステップＳ２５１２では、ステップＳ２５０８でリスト

アップした中でスタック位置確定フラグが暫定である通常ウィンドウ全てを、それらの中でみたスタック順はそのままで、内部データベースの中でスタック位置確定フラグが確定済みであるスタック最上位の通常ウィンドウの直上となるよう、ウィンドウシステム１２にスタック変更要求を発行し、ステップＳ２５１４に進む。

ステップＳ２５１２のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図を図２７に示す。

図２７に示されるように、マップ対象ウィンドウ及びマップ対象ウィンドウと同じグループ値を持つスタック位置確定フラグが暫定の通常ウィンドウが、スタック位置確定フラグが確定済みの通常ウィンドウ全体のスタック最上位ウィンドウの直上に移されているのが解る。

図２５において、ステップＳ２５１３では、ステップＳ２５０８でリストアップした中でスタック位置確定フラグが暫定である通常ウィンドウ全てを、それらの中でみたスタック順はそのままで、全体スタックの最下位となるよう、ウィンドウシステム１２にスタック変更要求を発行し、ステップＳ２５１４に進む。

ステップＳ２５１３のスタック変更要求によってスタックがどのように変化するかを示す具体例となる模式図を図２８に示す。

図２８に示されるように、マップ対象ウィンドウ及びマップ対象ウィンドウと同じグループ値を持つスタック位置確定フラグが暫定の通常ウィンドウが、スタック最下位に

移されているのが解る。

図２５において、ステップＳ２５１４では、ウィンドウシステム１２に対してマップ対象ウィンドウのマップ要求を発行し、内部データベースの対象ウィンドウのウィンドウ情報の内、スタック位置確定フラグが未確定を示している場合は確定済みに更新し、また、内部データベース全体に対して必要に応じてスタック位置を更新した上で、ステップＳ２５１５に進む。

ステップＳ２５１５では、処理を終了する。

以上のことにより、ウィンドウ管理プログラムが、マップ対象ウィンドウと同じグループ値を持つスタック位置が暫定の通常ウィンドウ群を、その通常ウィンドウ群の中でのスタック上下関係は保持しつつスタック位置を移してグループ毎に纏めるため、マップによって、同一グループ内でみたスタック順は変更されず、アプリケーションの同一グループ内のスタック順管理の負担が軽減される。更に、実施の形態１で示したように、スタック変更要求受信時にも、対象ウィンドウと同じグループ値を持つスタック位置が暫定の通常ウィンドウ群を、その通常ウィンドウ群の中でのスタック上下関係は保持しつつグループ毎に纏める事で、アプリケーションプログラムは、自グループに属するウィンドウのスタック順をウィンドウの生成順を基本として管理する事ができる。これは、Ｘウィンドウシステムの基本と一致しており、アプリケーションプログラムは、自グループの通常ウィンドウについて、一般的なスタック順決定ルールである生成順ベースで重なりを管理すれば良く

なる。

なお、本実施の形態では、マップ要求イベント（Ｍａｐ Ｒｅｑｕｅｓｔ）受信時に、ウィンドウマネージャのウィンドウのグループ値取得を行っているが、これに限定されるものではなく、実施の形態１と同様、アプリケーションプログラムがウィンドウ種類及びグループ設定をしたタイミングでグループ値を取得して内部データベースにウィンドウ種類とグループ値とプロパティ獲得済みである事を記憶しておいても良く、その方がむしろ処理効率が良い。しかし、実施の形態１と違い、前記タイミングで、スタック順をグループ毎に纏めてはいけない。

産業上の利用可能性

本発明は、表示装置に複数のウィンドウを表示するのに好適であり、特に、画面が小さく、結果として、ウィンドウの重なり易い携帯電話やＰＤＡ等におけるウィンドウの管理に好適である。

請求の範囲

1．1つ以上のアプリケーションプログラムに基づいて表示装置に表示される複数のウィンドウの重ね合わせを管理するためのウィンドウスタック制御方法であって、

アプリケーションプログラムからウィンドウのグループの指定を受けるステップと、

アプリケーションプログラムからウィンドウの表示要求を受けるステップと、

表示要求を受けて当該ウィンドウの表示を行う際に、当該ウィンドウのスタック順をグループ毎に一連となるように纏めるステップとを備えたウィンドウスタック制御方法。

2．前記纏めるステップは、グループ毎に纏めていない第1のウィンドウの表示要求をアプリケーションプログラムから受け取った際に、グループ毎に纏めていないウィンドウの中で前記第1のウィンドウと同じグループに属する第1のウィンドウ群のスタック順が、該第1のウィンドウ群の中でのスタック順関係は保持したまま、既にグループ毎に纏められたウィンドウの中で前記第1のウィンドウと同じグループに属する第2のウィンドウ群のスタック順と一連となるようにすることを特徴とする、請求項1に記載のウィンドウスタック制御方法。

3．前記ウィンドウスタック制御方法は、グループ毎に代表ウィンドウを1枚生成するステップをさらに備え、

前記纏めるステップは、ウィンドウのスタック順をグル

ープ毎に一連とする際に、当該ウィンドウを前記代表ウィンドウの子ウィンドウとする事を特徴とする、請求項１または２に記載のウィンドウスタック制御方法。

４．アプリケーションプログラムからグループ内でのウィンドウスタックの最上位移動要求および最下位移動要求を受けるステップと、

前記最上位移動要求または前記最下位移動要求を受けて、グループ内でのスタック変更を行うステップとをさらに備えた、請求項１から３のいずれかに記載のウィンドウスタック制御方法。

５．前記スタック変更を行うステップは、前記第１のウィンドウに対するグループ内でのウィンドウスタックの最上位移動要求および最下位移動要求をアプリケーションプログラムから受け取った際に、グループ毎に纏めていないウィンドウの中で前記第１のウィンドウと同じグループに属する第１のウィンドウ群のスタック順が、前記第１のウィンドウ群の中でのスタック順関係は保持したまま、既にグループ毎に纏められたウィンドウの中で前記第１のウィンドウと同じグループに属する第２のウィンドウ群のスタック順と一連となるようにスタック変更を行うことを特徴とする、請求項４に記載のウィンドウスタック制御方法。

６．アプリケーションプログラムからグループ単位でのウィンドウスタックの最上位移動要求および最下位移動要求を受けるステップと、

前記最上位移動要求または前記最下位移動要求を受けて、グループ単位でのスタック変更を行うステップとをさら

に備えた、請求項１から３のいずれかに記載のウィンドウスタック制御方法。

７．前記スタック変更を行うステップは、第１のグループに対するグループ単位でのウィンドウスタックの最上位移動要求および最下位移動要求をアプリケーションプログラムから受け取った際に、グループ毎に纏めていないウィンドウの中で前記第１のグループに属する第１のウィンドウ群のスタック順が、前記第１のウィンドウ群の中でのスタック順関係は保持したまま、既にグループ毎に纏められたウィンドウの中で前記第１のグループに属する第２のウィンドウ群のスタック順と一連となるようにスタック変更を行うことを特徴とする、請求項５に記載のウィンドウスタック制御方法。

８．アプリケーションプログラムによって優先ウィンドウとして指定されたウィンドウをいずれのグループにも所属させず、かつ当該優先ウィンドウを、表示装置に表示された、いずれかのグループに所属する全てのウィンドウより常にスタック上位に位置させることを特徴とする、請求項１に記載のウィンドウスタック制御方法。

９．最上位グループに属するウィンドウよりスタック下位のウィンドウを常に非表示状態とすることを特徴とする、請求項１に記載のウィンドウスタック制御方法。

１０．最上位グループに属するウィンドウの中のスタック最下位のウィンドウの直下に、特定のウィンドウを位置させることを特徴とする、請求項１に記載のウィンドウスタック制御方法。

１１．前記ウィンドウスタック制御方法は、Ｘウィンドウシステムとウィンドウマネージャによって複数のウィンドウの重ね合わせを管理するものであって、

最上位グループの直上に特定のウィンドウを位置させることを特徴とする、請求項４または６に記載のウィンドウスタック制御方法。

１２．前記ウィンドウスタック制御方法は、Ｘウィンドウシステムとウィンドウマネージャによって複数のウィンドウの重ね合わせを管理するものであって、

ウィンドウマネージャが、ウィンドウ破棄通知受信時に、ウィンドウシステム間とでスタック認識の整合性を確認し、異なっている場合は、ウィンドウシステムのスタック認識をウィンドウマネージャのスタック認識に合わせ込む処理を行うことを特徴とする、請求項１から３のいずれかに記載のウィンドウスタック制御方法。

１３．前記ウィンドウスタック制御方法は、Ｘウィンドウシステムとウィンドウマネージャによって複数のウィンドウの重ね合わせを管理するものであって、

ウィンドウマネージャが、ウィンドウシステムにスタック変更を要求する際にフラグを立てる一方で、ウィンドウ破棄通知受信時に、前記フラグが立っている時のみ、ウィンドウシステム間とでスタック認識の整合性を確認し、異なっている場合は、ウィンドウシステムのスタック認識をウィンドウマネージャのスタック認識に合わせ込む処理を行い、前記フラグを下げることを特徴とする、請求項１から３のいずれかに記載のウィンドウスタック制御方法。

１４．前記纏めるステップは、グループ毎に纏めていない第１のウィンドウの表示要求をアプリケーションプログラムから受け取った際に、既にグループ毎に纏められたウィンドウの中で前記第１のウィンドウと同じグループに属する第２のウィンドウ群のスタック順と一連となるようにすることを特徴とする、請求項１に記載のウィンドウスタック制御方法。

１５．１つ以上のアプリケーションプログラムに基づいて表示装置に表示される複数のウィンドウの重ね合わせを管理するためのウィンドウ管理プログラムであって、コンピュータに、

アプリケーションプログラムからウィンドウのグループの指定を受けるステップと、

アプリケーションプログラムからウィンドウの表示要求を受けるステップと、

表示要求を受けた際に該ウィンドウのスタック順をグループ毎に一連となるように纏めるステップとを実行させるためのウィンドウ管理プログラム。

１６．表示装置に複数のウィンドウを表示する際のウィンドウの重ね合わせを管理するウィンドウ管理装置であって、

１つ以上のウィンドウを前記表示装置に表示する１つ以上のアプリケーションプログラムと、

前記１つ以上のアプリケーションプログラムが表示するウィンドウの重ね合わせを管理するウィンドウ管理プログラムと、

前記アプリケーションプログラムおよび前記ウィンドウ管理プログラムを実行する処理部とを備え、

前記アプリケーションプログラムは、前記ウィンドウ管理プログラムに対してウィンドウのグループを指定するものであり、

前記ウィンドウ管理プログラムは、前記アプリケーションプログラムからウィンドウの表示要求を受けて当該ウィンドウの表示を行う際に、当該ウィンドウのスタック順をグループ毎に一連とする制御を行うものであることを特徴とする、ウィンドウ管理装置。

補正書の請求の範囲

[２００４年９月３日（０３．０９．２００４）国際事務局受理：出願当初の請求の範囲２は取り下げられた；出願当初の請求の範囲１は補正された；他の請求の範囲は変更なし。（１頁）]

１．（補正後）１つ以上のアプリケーションプログラムに基づいて表示装置に表示される複数のウィンドウの重ね合わせを管理するためのウィンドウスタック制御方法であって、

アプリケーションプログラムからウィンドウの新規作成要求を受けるステップと、

前記アプリケーションプログラムから前記ウィンドウのグループの指定を受けるステップと、

前記アプリケーションプログラムから前記ウィンドウの表示要求を受けるステップと、

前記表示要求を受けて当該ウィンドウの表示を行う際に、当該ウィンドウのスタック順をグループ毎に一連となるように纏めるステップと、を備え、

前記纏めるステップは、

既に一連となるように纏められたウィンドウ群におけるグループ間のスタック順関係は保持したまま、前記ウィンドウのスタック順を前記ウィンドウの属するグループと一連となるように制御する

ことを特徴とするウィンドウスタック制御方法。

２．（削除）

３．前記ウィンドウスタック制御方法は、グループ毎に代表ウィンドウを１枚生成するステップをさらに備え、

前記纏めるステップは、ウィンドウのスタック順をグル

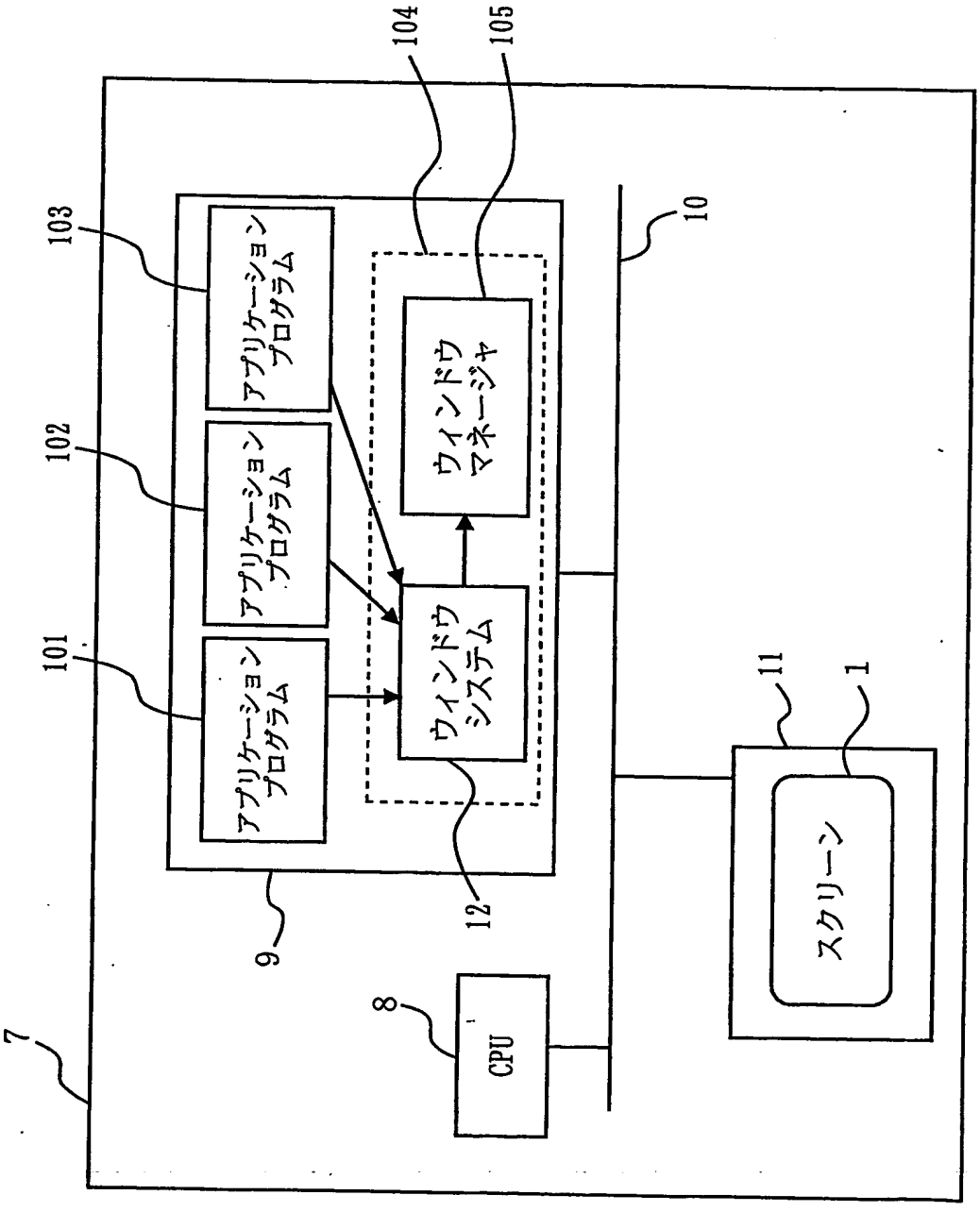

104
105
103
10
アプリケーション プログラム
ウィンドウ マネージャ
102
アプリケーション プログラム
101
アプリケーション プログラム
ウィンドウ システム
11
1
スクリーン
9
12
8
CPU
7

図1

図2

情報名	内容
ウィンドウ識別子	識別子
スタック位置確定フラグ	暫定／確定済み
スタック位置情報	位置情報
プロパティ獲得フラグ	未獲得／獲得済み
ウィンドウ種類	優先ウィンドウ／通常ウィンドウ
グループ値	グループの値

図３

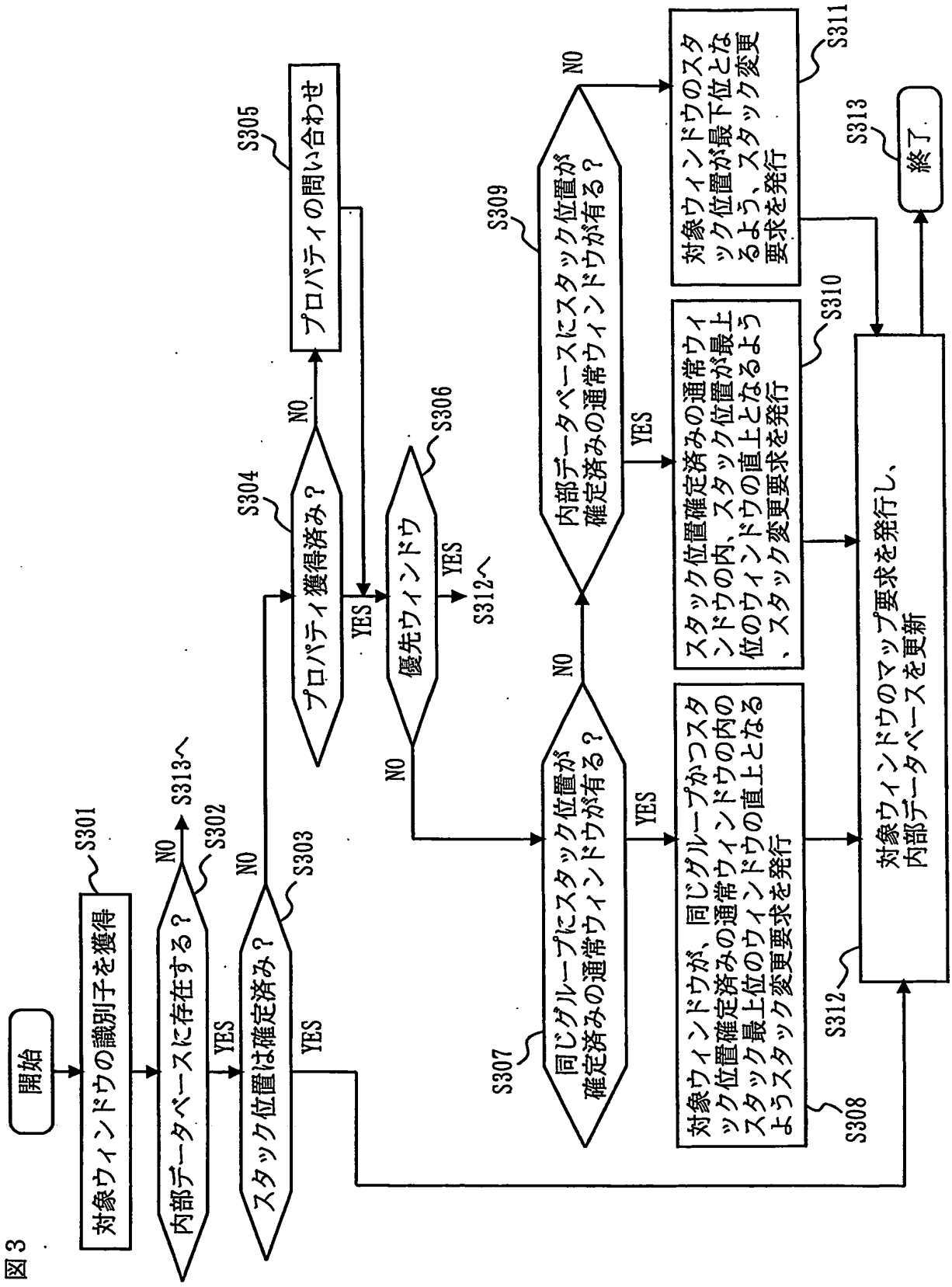

開始
S301
対象ウィンドウの識別子を獲得
S302
内部データベースに存在する？
NO
S313へ
YES
S303
スタック位置は確定済み？
NO
YES
S304
プロパティ獲得済み？
NO
S305
プロパティの問い合わせ
YES
S306
優先ウィンドウ
NO
YES
S312へ
S307
同じグループにスタック位置が確定済みの通常ウィンドウが有る？
NO
YES
S309
内部データベースにスタック位置が確定済みの通常ウィンドウが有る？
NO
YES
対象ウィンドウが、同じグループかつスタック位置確定済みの内のスタック最上位のウィンドウの直上となるようスタック変更要求を発行
スタック位置確定済みの通常ウィンドウの内、スタック位置が最上位のウィンドウの直上となるよう、スタック変更要求を発行
S310
対象ウィンドウのスタック位置が最下位となるよう、スタック変更要求を発行
S311
S308
S312
対象ウィンドウのマップ要求を発行し、内部データベースを更新
S313
終了

図4

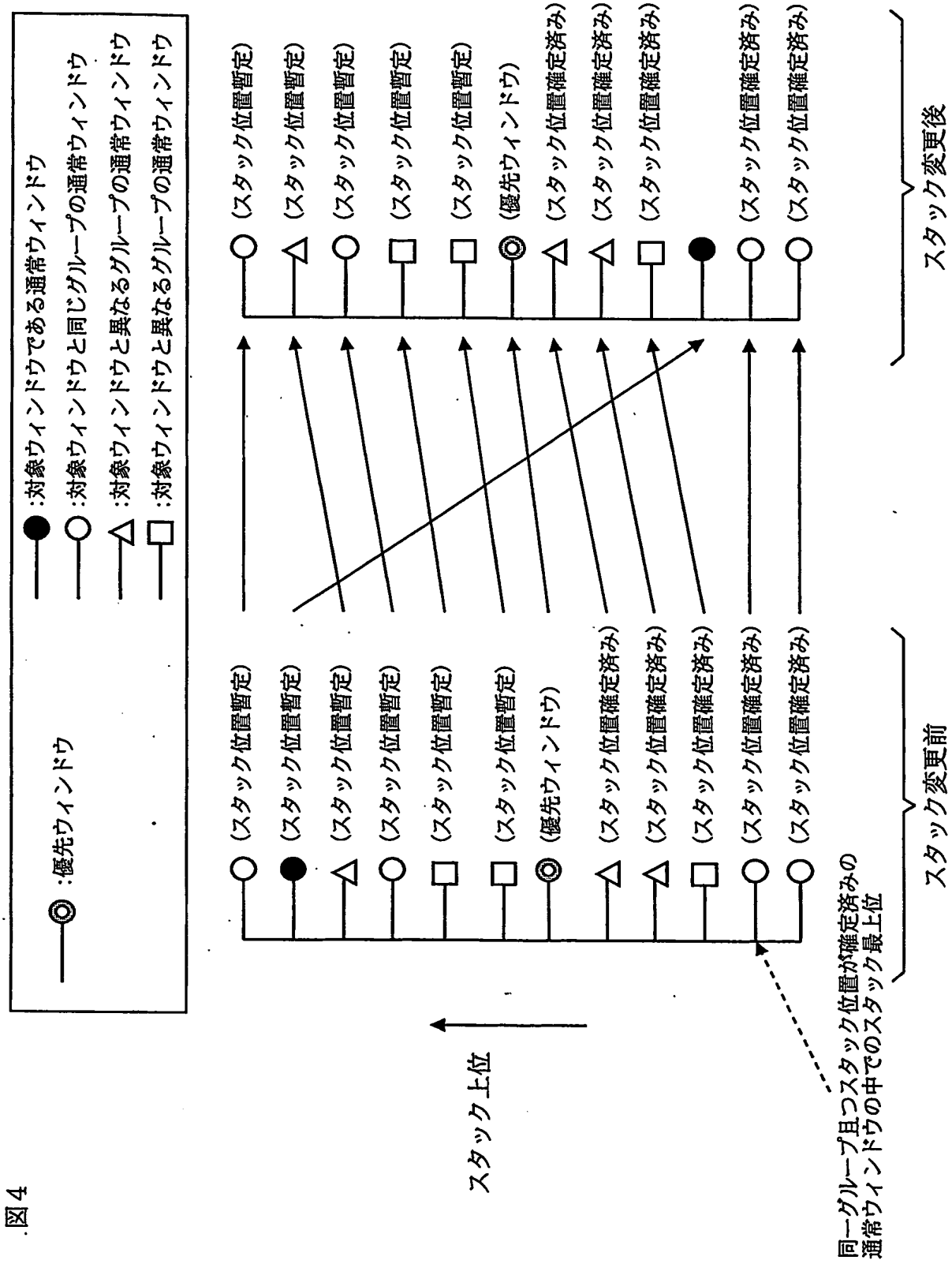

:対象ウィンドウである通常ウィンドウ
:対象ウィンドウと同じグループの通常ウィンドウ
:対象ウィンドウと異なるグループの通常ウィンドウ
:対象ウィンドウと異なるグループの通常ウィンドウ
:優先ウィンドウ
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(優先ウィンドウ)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
スタック上位
同一グループ且つスタック位置が確定済みの
通常ウィンドウの中でのスタック最上位
スタック変更前
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(優先ウィンドウ)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
スタック変更後

図5

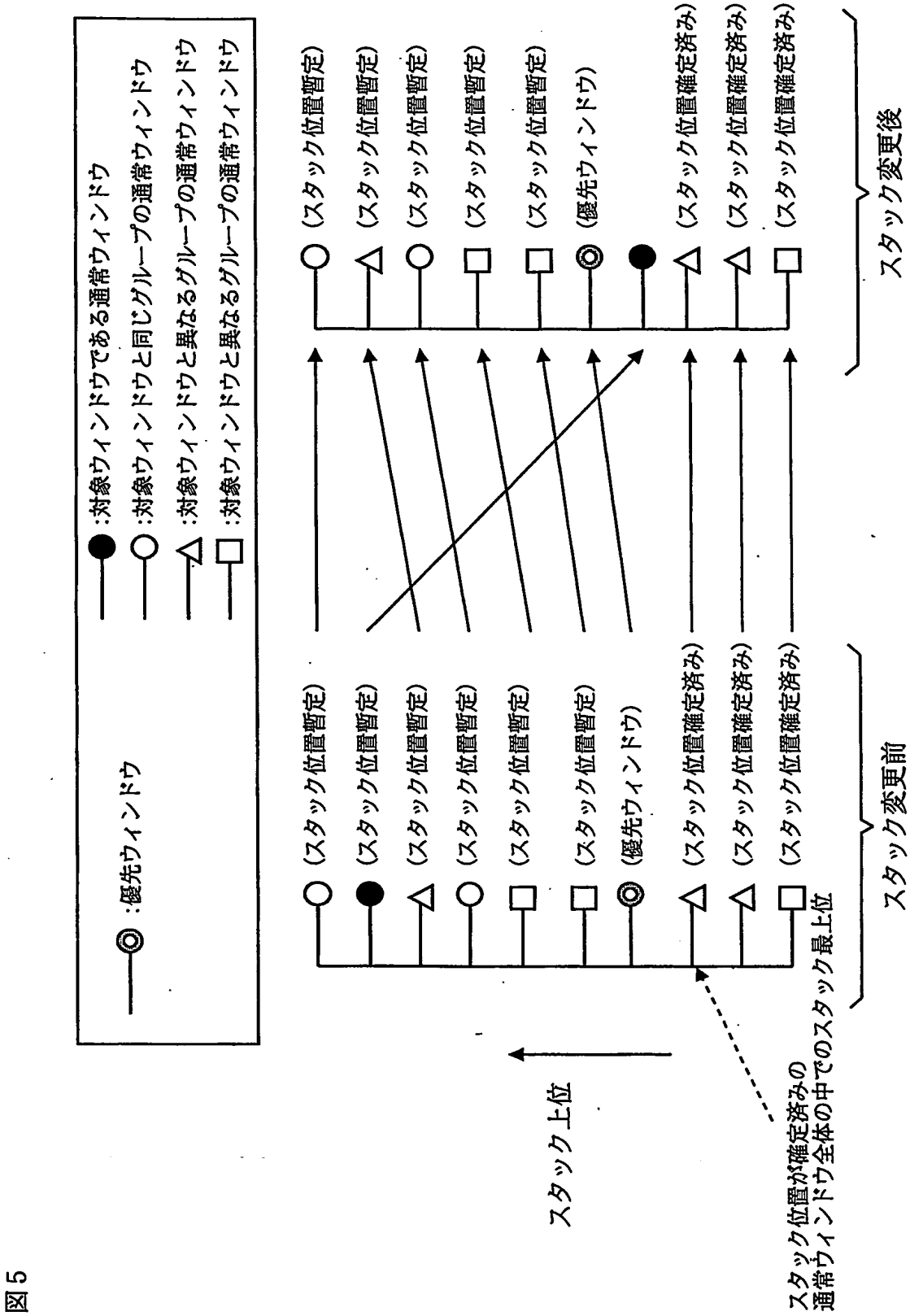

:対象ウィンドウである通常ウィンドウ
:対象ウィンドウと同じグループの通常ウィンドウ
:対象ウィンドウと異なるグループの通常ウィンドウ
:対象ウィンドウと異なるグループの通常ウィンドウ
:優先ウィンドウ
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(優先ウィンドウ)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
スタック上位
スタック位置が確定済みの
通常ウィンドウ全体の中でのスタック最上位
スタック変更前
スタック変更後

図6

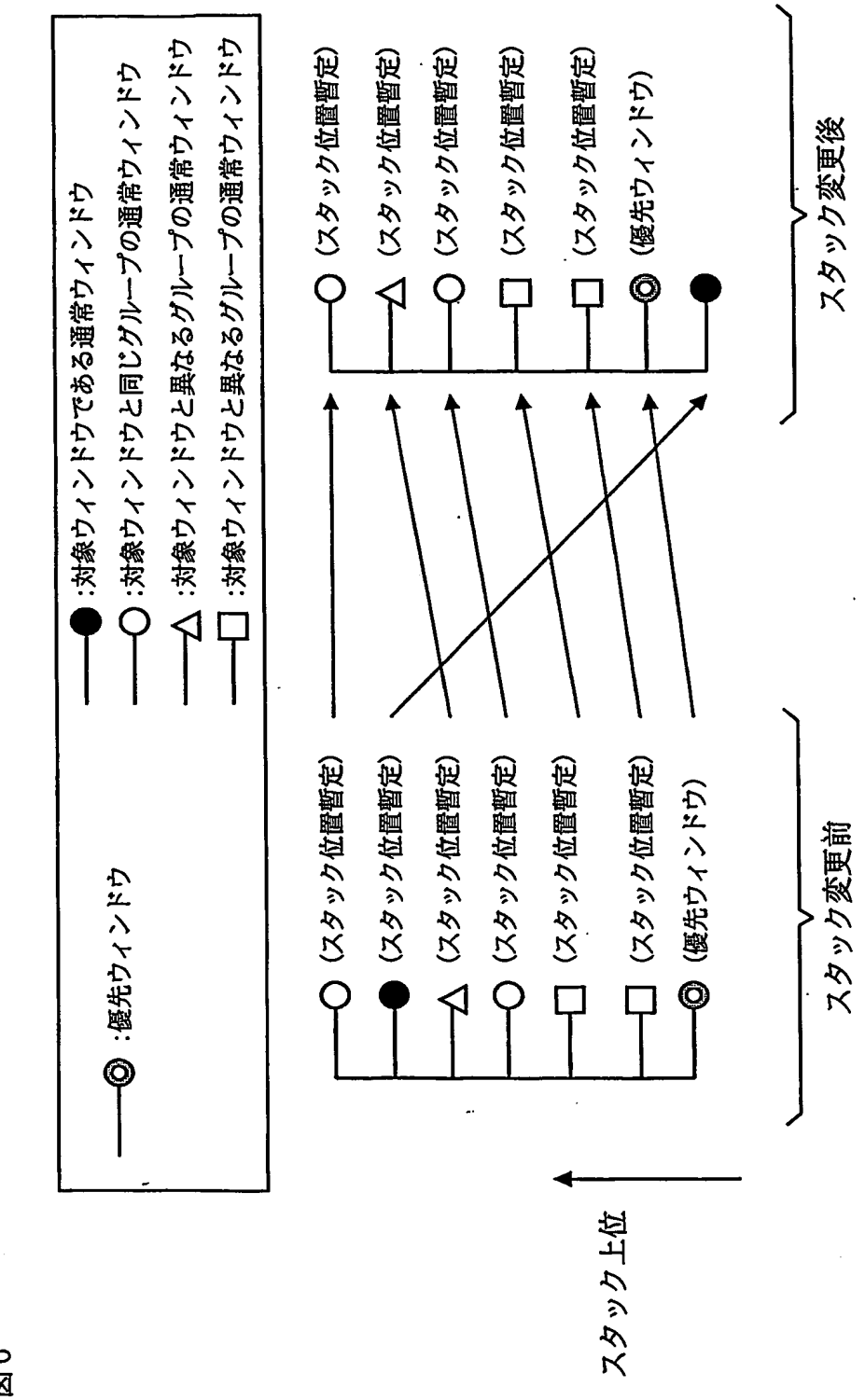

:対象ウィンドウである通常ウィンドウ
:対象ウィンドウと同じグループの通常ウィンドウ
:対象ウィンドウと異なるグループの通常ウィンドウ
:対象ウィンドウと異なるグループの通常ウィンドウ
:優先ウィンドウ
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(優先ウィンドウ)
スタック上位
スタック変更前
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(優先ウィンドウ)
スタック変更後

図7

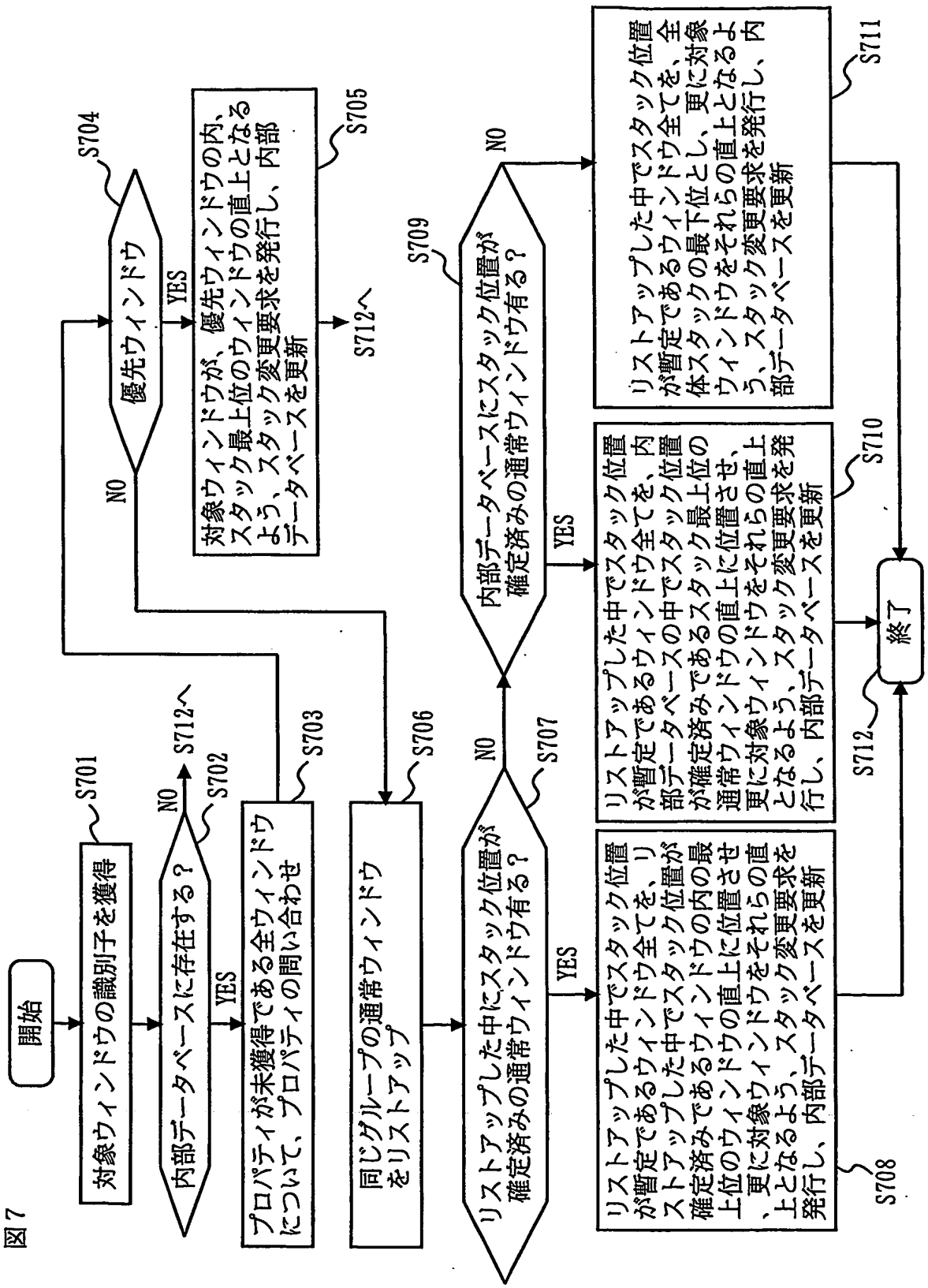

開始
S701
対象ウィンドウの識別子を獲得
S702
内部データベースに存在する？
NO
S712へ
YES
S703
プロパティが未獲得である全ウィンドウについて、プロパティの問い合わせ
S704
優先ウィンドウ
NO
YES
S705
対象ウィンドウが、優先ウィンドウの内、スタック最上位のウィンドウの直上となるよう、スタック変更要求を発行し、内部データベースを更新
S712へ
S706
同じグループの通常ウィンドウをリストアップ
S707
リストアップした中にスタック位置が確定済みの通常ウィンドウ有る？
NO
YES
S708
リストアップした中でスタック位置が暫定であるウィンドウ全てを、リストアップした中でスタック位置が確定済みであるウィンドウの内の最上位のウィンドウの直上に位置させ、更に対象ウィンドウをそれらの直上となるよう、スタック変更要求を発行し、内部データベースを更新
S709
内部データベースにスタック位置が確定済みの通常ウィンドウ有る？
NO
YES
S710
リストアップした中でスタック位置が暫定であるウィンドウ全てを、内部データベースの中でスタック位置が確定済みであるスタック最上位の通常ウィンドウの直上に位置させ、更に対象ウィンドウをそれらの直上となるよう、スタック変更要求を発行し、内部データベースを更新
S711
リストアップした中でスタック位置が暫定であるウィンドウ全てを、全体スタックの最下位とし、更に対象ウィンドウをそれらの直上となるよう、スタック変更要求を発行し、内部データベースを更新
S712
終了

図8

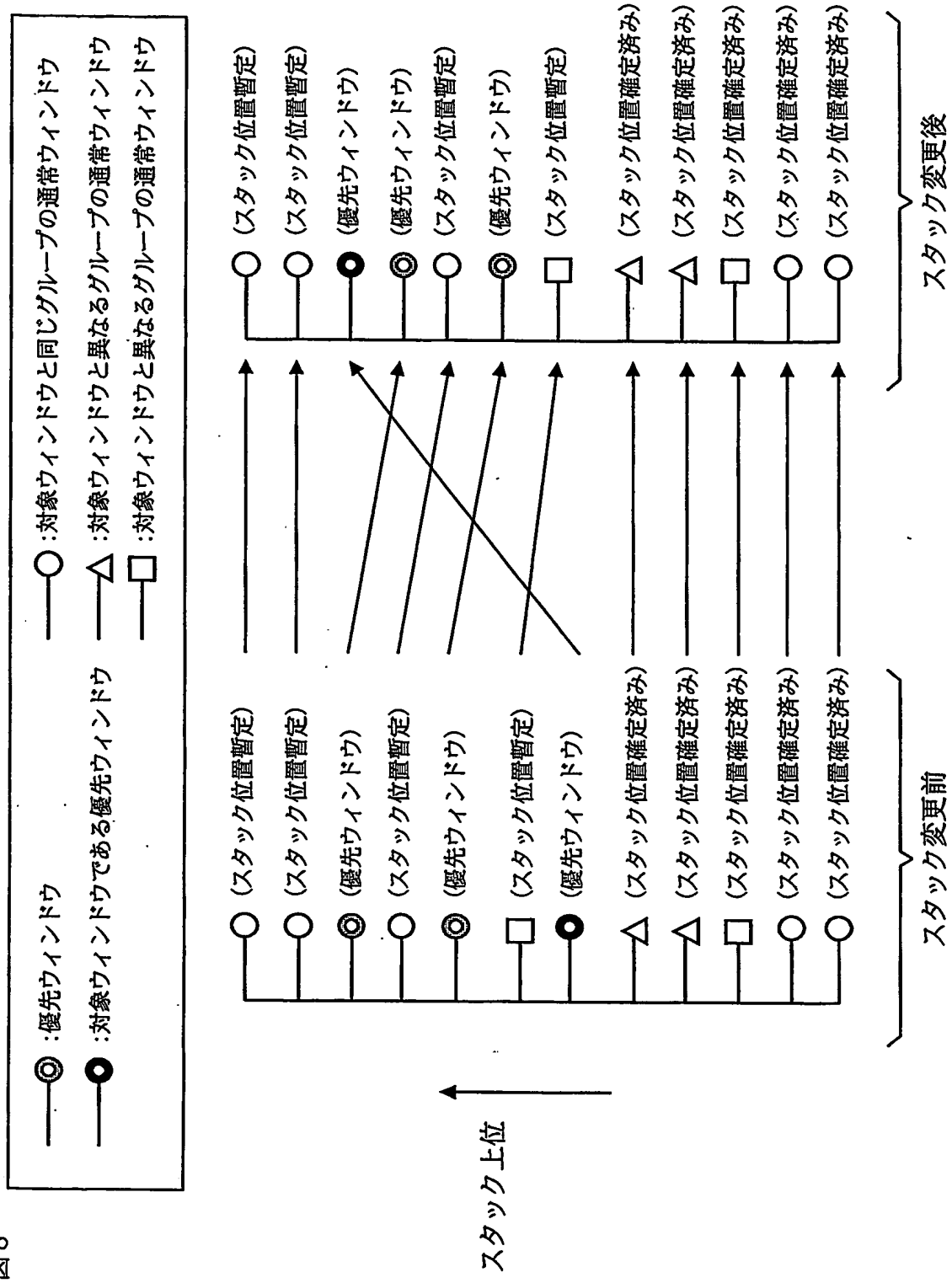

:優先ウィンドウ
:対象ウィンドウと同じグループの通常ウィンドウ
:対象ウィンドウである優先ウィンドウ
:対象ウィンドウと異なるグループの通常ウィンドウ
:対象ウィンドウと異なるグループの通常ウィンドウ
スタック上位
(スタック位置暫定)
(スタック位置暫定)
(優先ウィンドウ)
(スタック位置暫定)
(優先ウィンドウ)
(スタック位置暫定)
(優先ウィンドウ)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
スタック変更前
(スタック位置暫定)
(スタック位置暫定)
(優先ウィンドウ)
(優先ウィンドウ)
(スタック位置暫定)
(優先ウィンドウ)
(スタック位置暫定)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
スタック変更後

図9

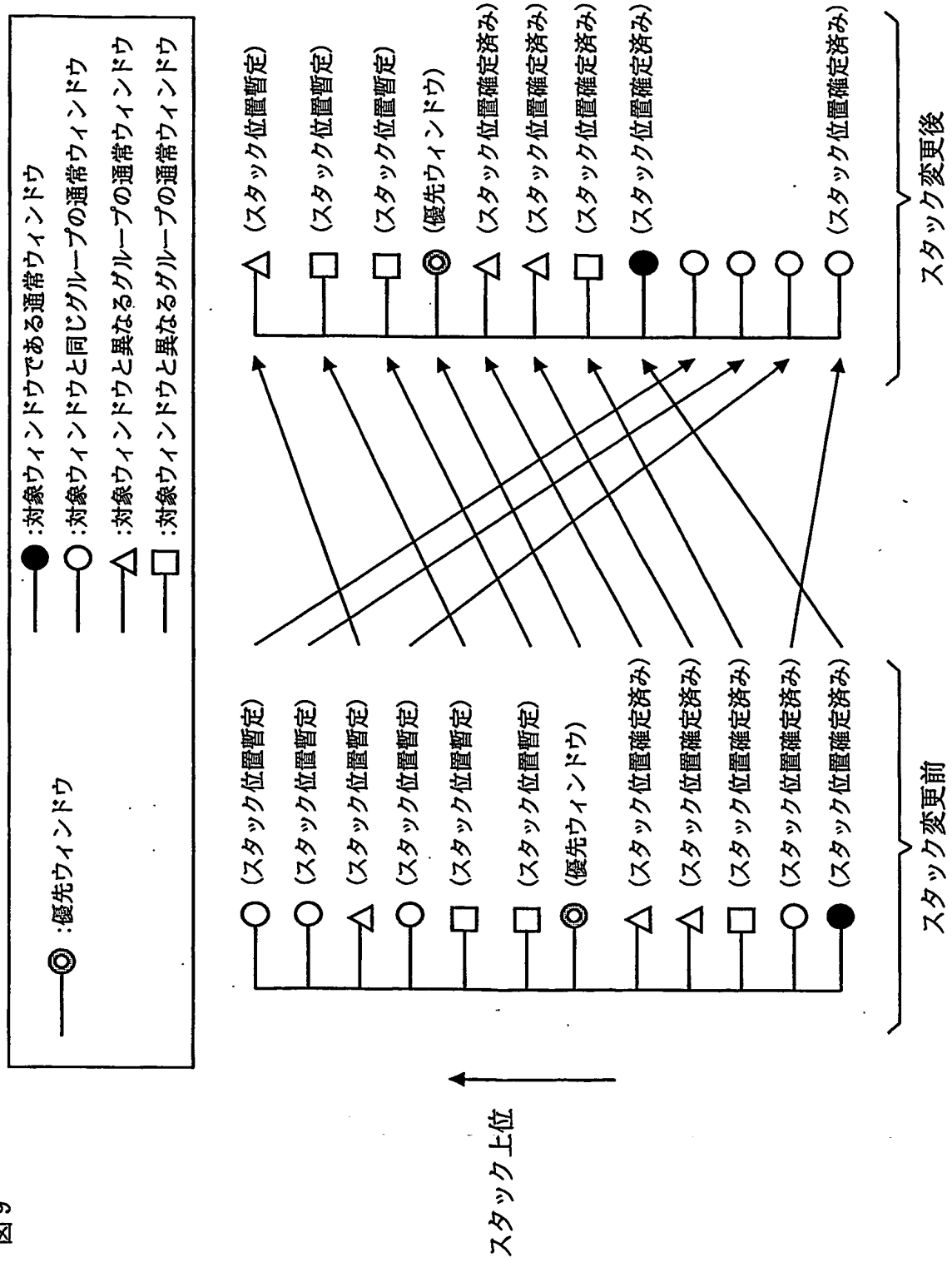

:対象ウィンドウである通常ウィンドウ
:対象ウィンドウと同じグループの通常ウィンドウ
:対象ウィンドウと異なるグループの通常ウィンドウ
:対象ウィンドウと異なるグループの通常ウィンドウ
:優先ウィンドウ
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(優先ウィンドウ)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
スタック上位
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
スタック変更前
スタック変更後

図１０

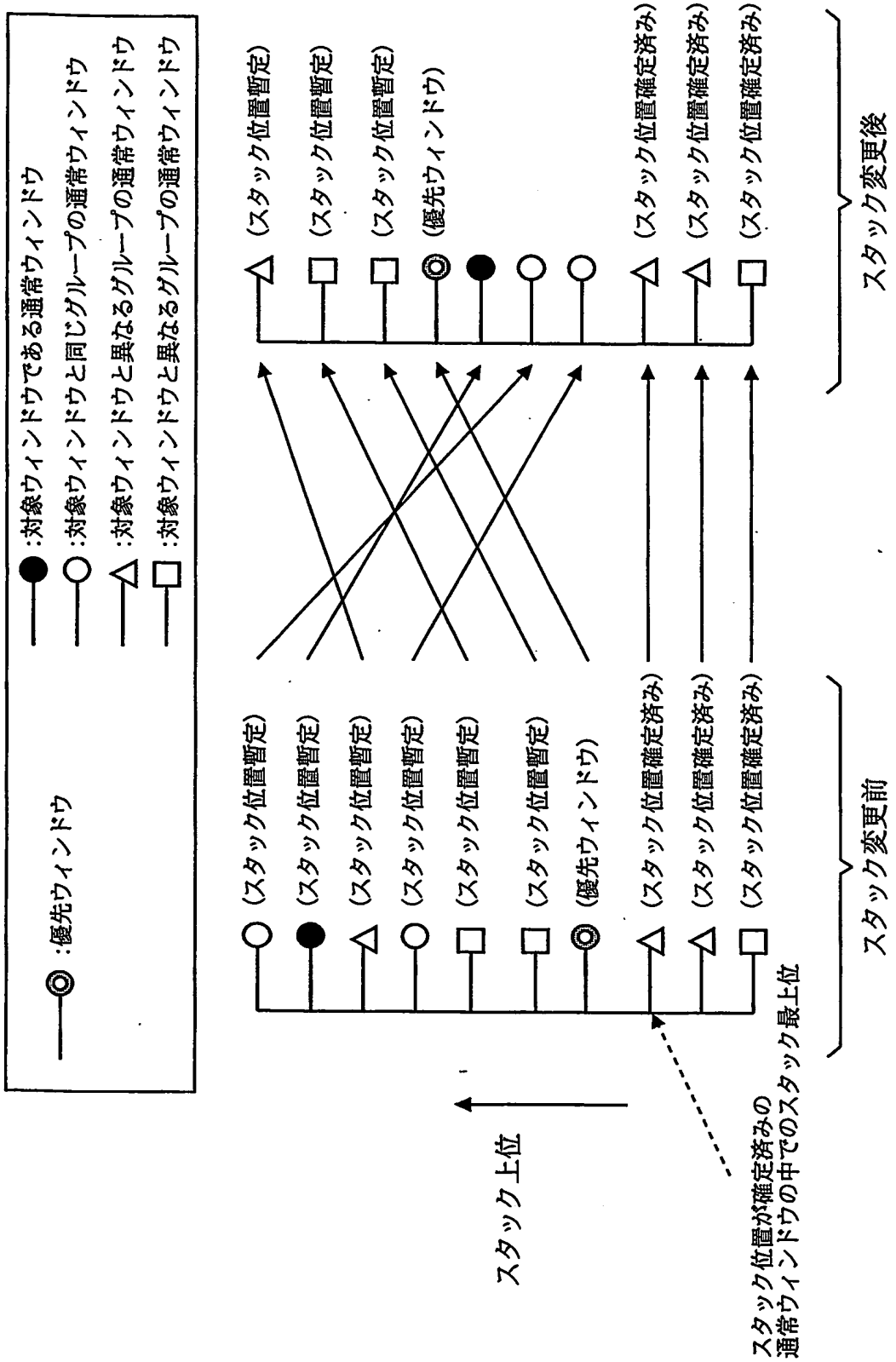

:対象ウィンドウである通常ウィンドウ
:対象ウィンドウと同じグループの通常ウィンドウ
:対象ウィンドウと異なるグループの通常ウィンドウ
:対象ウィンドウと異なるグループの通常ウィンドウ
:優先ウィンドウ
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(優先ウィンドウ)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
スタック上位
スタック位置が確定済みの
通常ウィンドウの中でのスタック最上位
スタック変更前
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(優先ウィンドウ)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
スタック変更後

図１１

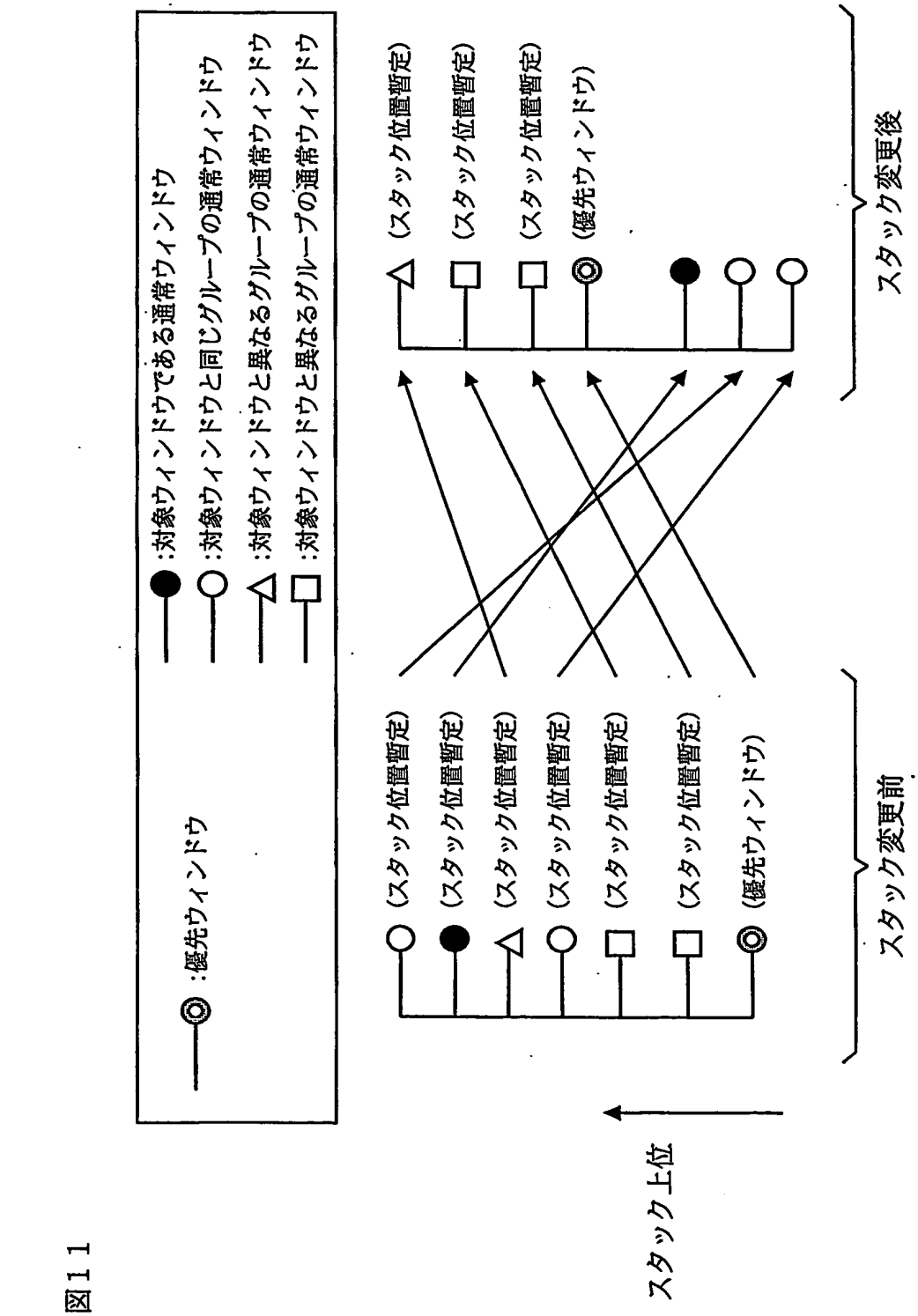

:優先ウィンドウ
:対象ウィンドウである通常ウィンドウ
:対象ウィンドウと同じグループの通常ウィンドウ
:対象ウィンドウと異なるグループの通常ウィンドウ
:対象ウィンドウと異なるグループの通常ウィンドウ
スタック上位
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(優先ウィンドウ)
スタック変更前
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(優先ウィンドウ)
スタック変更後

図１２

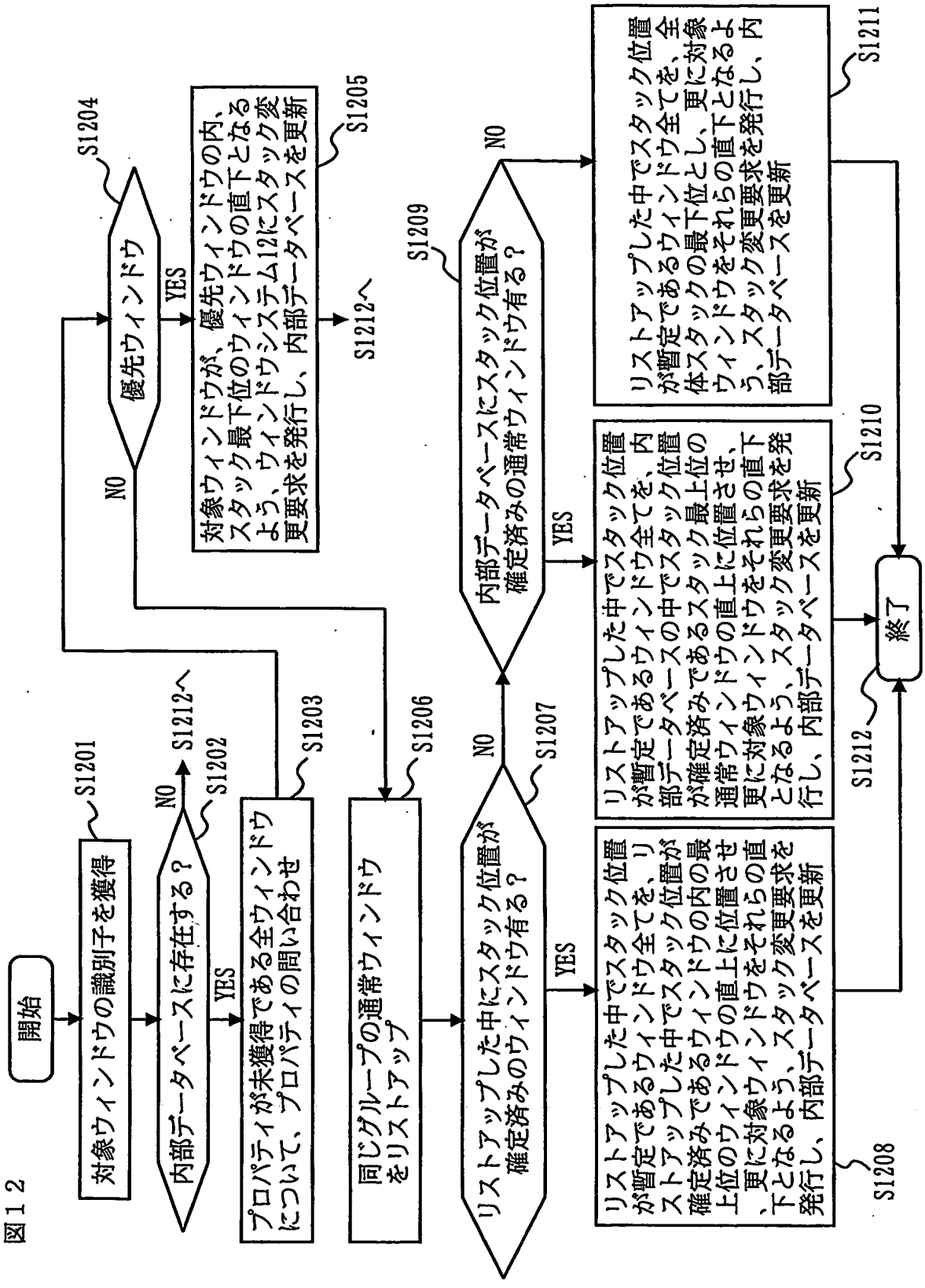

開始
S1201
対象ウィンドウの識別子を獲得
S1202
内部データベースに存在する？
NO
S1212へ
YES
S1203
プロパティが未獲得である全ウィンドウについて、プロパティの問い合わせ
S1204
優先ウィンドウ
NO
YES
S1205
対象ウィンドウが、優先ウィンドウの内、スタック最下位のウィンドウの直下となるよう、ウィンドウシステム12にスタック変更要求を発行し、内部データベースを更新
S1212へ
S1206
同じグループの通常ウィンドウをリストアップ
S1207
リストアップした中にスタック位置が確定済みのウィンドウ有る？
NO
YES
S1208
リストアップした中でスタック位置が暫定であるウィンドウ全てを、リストアップした中でスタック位置が確定済みであるウィンドウの内の最上位のウィンドウの直上に位置させ、更に対象ウィンドウをそれらの直下となるよう、スタック変更要求を発行し、内部データベースを更新
S1209
内部データベースにスタック位置が確定済みの通常ウィンドウ有る？
NO
YES
S1210
リストアップした中でスタック位置が暫定であるウィンドウ全てを、内部データベースの中でスタック位置が確定済みであるスタック最上位の通常ウィンドウの直上に位置させ、更に対象ウィンドウをそれらの直下となるよう、スタック変更要求を発行し、内部データベースを更新
S1211
リストアップした中でスタック位置が暫定であるウィンドウ全てを、全体スタックの最下位とし、更に対象ウィンドウをそれらの直下となるよう、スタック変更要求を発行し、内部データベースを更新
S1212
終了

図１３

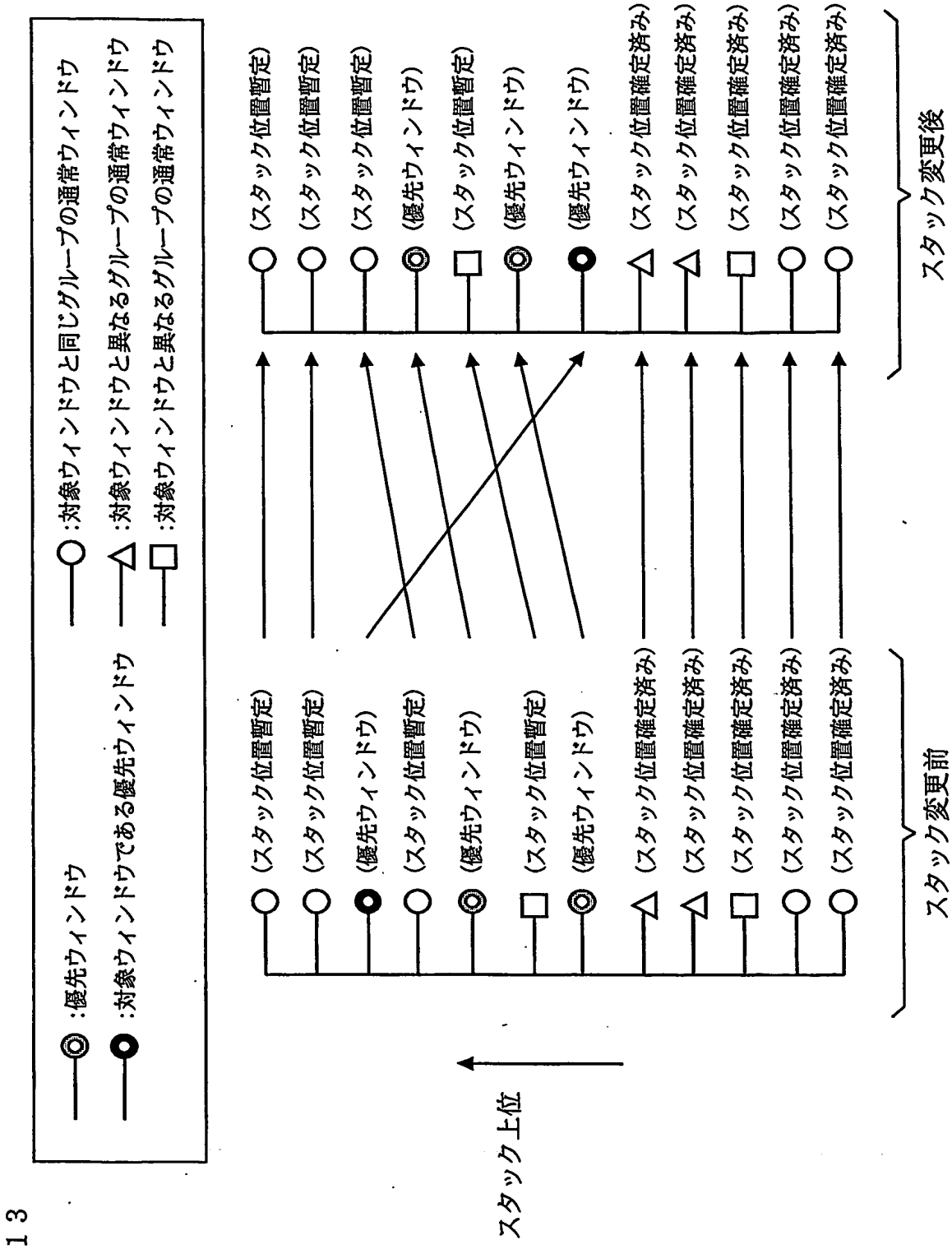

:優先ウィンドウ
:対象ウィンドウと同じグループの通常ウィンドウ
:対象ウィンドウである優先ウィンドウ
:対象ウィンドウと異なるグループの通常ウィンドウ
:対象ウィンドウと異なるグループの通常ウィンドウ
スタック上位
(スタック位置暫定)
(スタック位置暫定)
(優先ウィンドウ)
(スタック位置暫定)
(優先ウィンドウ)
(スタック位置暫定)
(優先ウィンドウ)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
スタック変更前
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(優先ウィンドウ)
(スタック位置暫定)
(優先ウィンドウ)
(優先ウィンドウ)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
スタック変更後

図１４

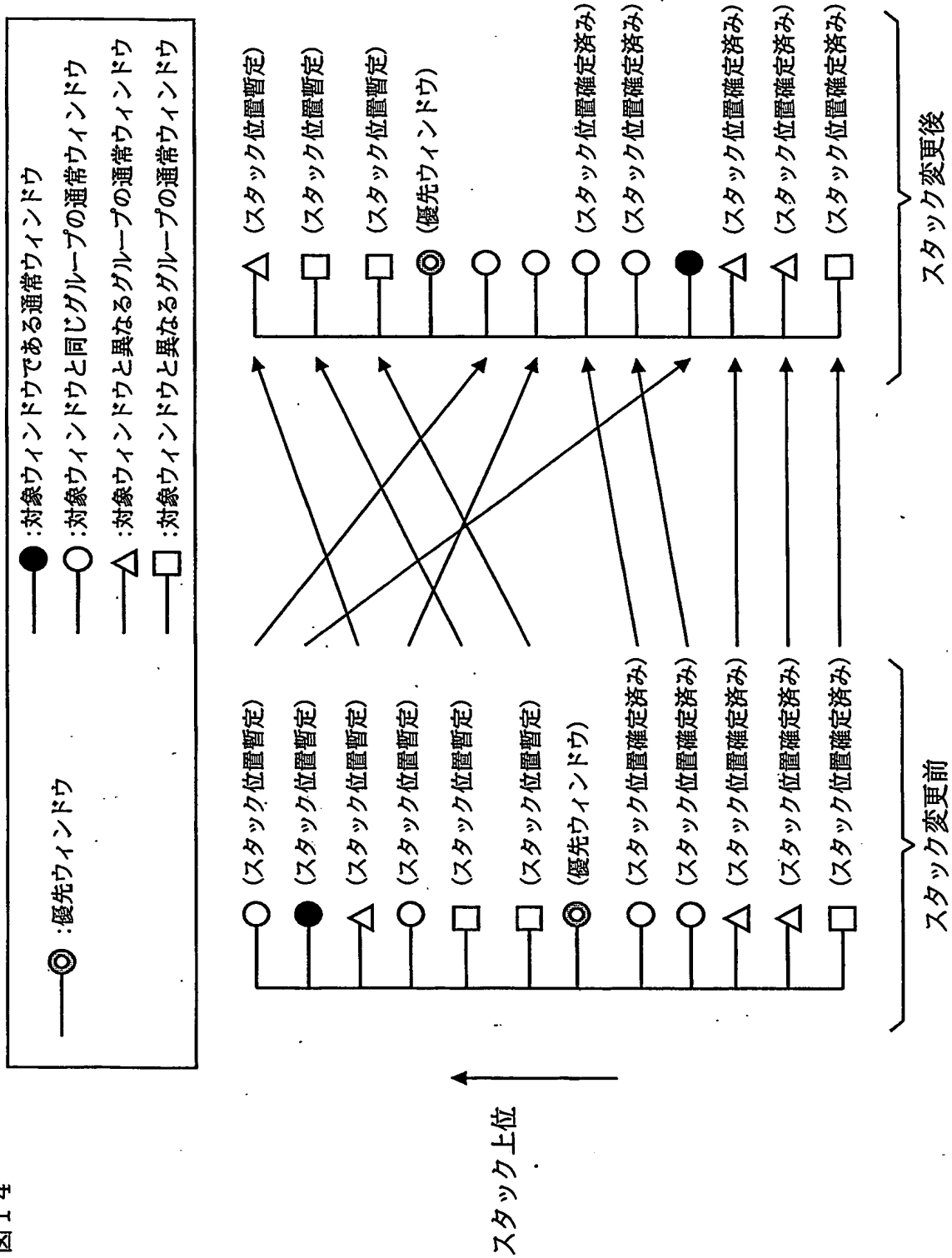

:対象ウィンドウである通常ウィンドウ
:対象ウィンドウと同じグループの通常ウィンドウ
:対象ウィンドウと異なるグループの通常ウィンドウ
:対象ウィンドウと異なるグループの通常ウィンドウ
:優先ウィンドウ
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(優先ウィンドウ)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
スタック上位
スタック変更前
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(優先ウィンドウ)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
スタック変更後

図15

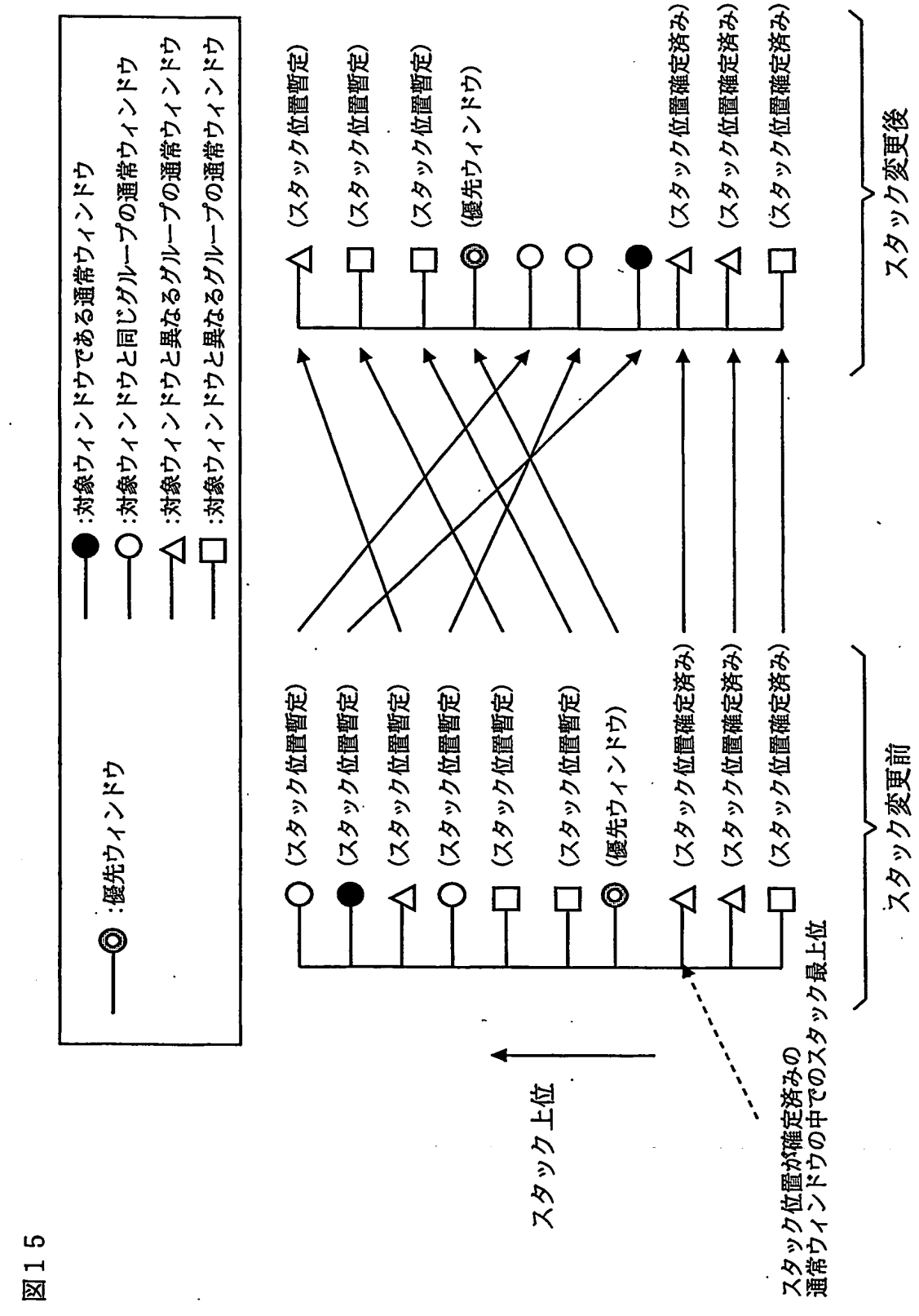

:対象ウィンドウである通常ウィンドウ
:対象ウィンドウと同じグループの通常ウィンドウ
:対象ウィンドウと異なるグループの通常ウィンドウ
:対象ウィンドウと異なるグループの通常ウィンドウ
:優先ウィンドウ
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(優先ウィンドウ)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
スタック上位
スタック位置が確定済みの
通常ウィンドウの中でのスタック最上位
スタック変更前
スタック変更後

図16

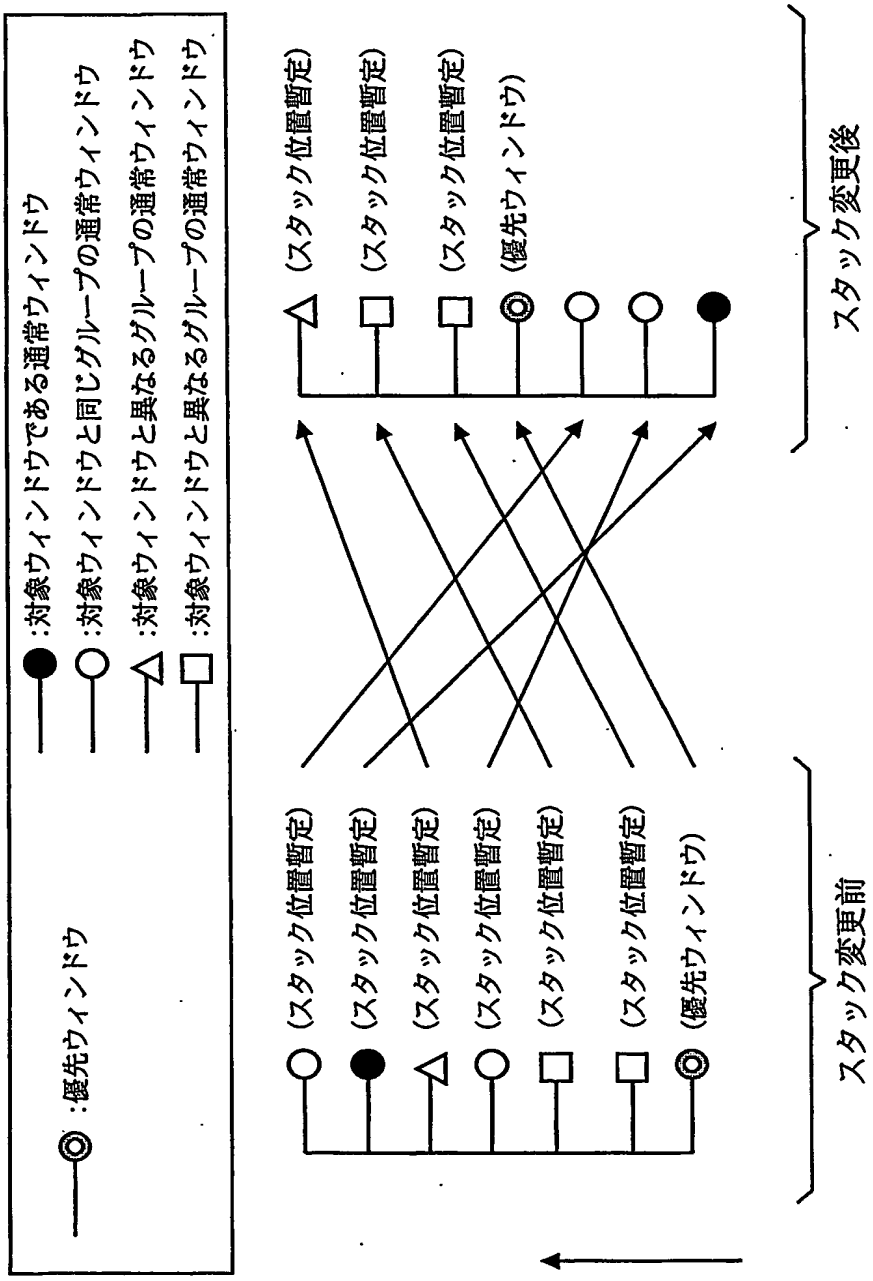

:対象ウィンドウである通常ウィンドウ
:対象ウィンドウと同じグループの通常ウィンドウ
:対象ウィンドウと異なるグループの通常ウィンドウ
:対象ウィンドウと異なるグループの通常ウィンドウ
:優先ウィンドウ
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(優先ウィンドウ)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(優先ウィンドウ)
スタック上位
スタック変更前
スタック変更後

図１７

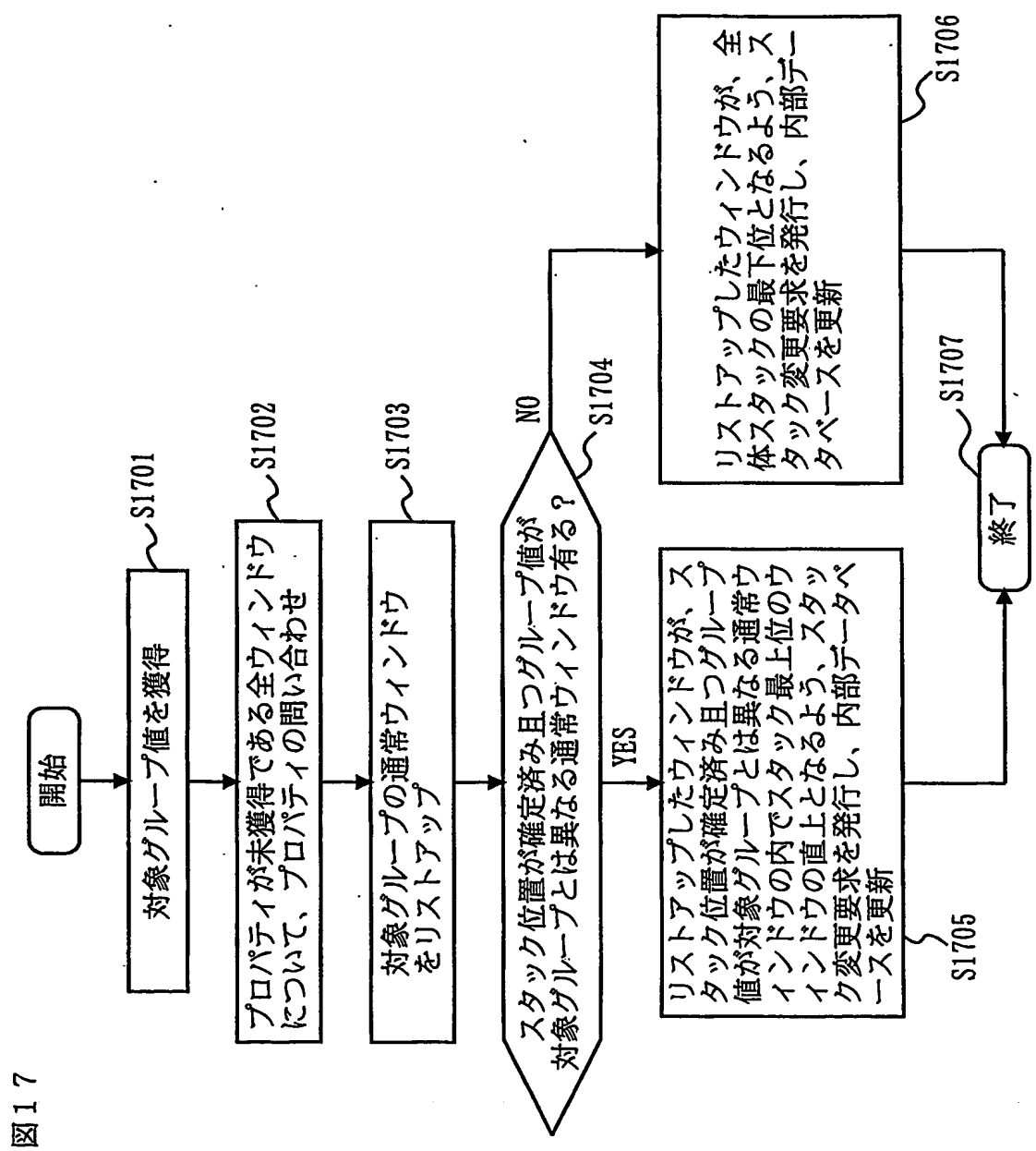

開始
対象グループ値を獲得
S1701
プロパティが未獲得である全ウィンドウについて、プロパティの問い合わせ
S1702
対象グループの通常ウィンドウをリストアップ
S1703
スタック位置が確定済み且つグループ値が対象グループとは異なる通常ウィンドウ有る？
S1704
NO
YES
リストアップしたウィンドウが、全体スタックの最下位となるよう、スタック変更要求を発行し、内部データベースを更新
S1706
リストアップしたウィンドウが、スタック位置が確定済み且つグループ値が対象グループとは異なる通常ウィンドウの内でスタック最上位のウィンドウの直上となるよう、スタック変更要求を発行し、内部データベースを更新
S1705
S1707
終了

図１８

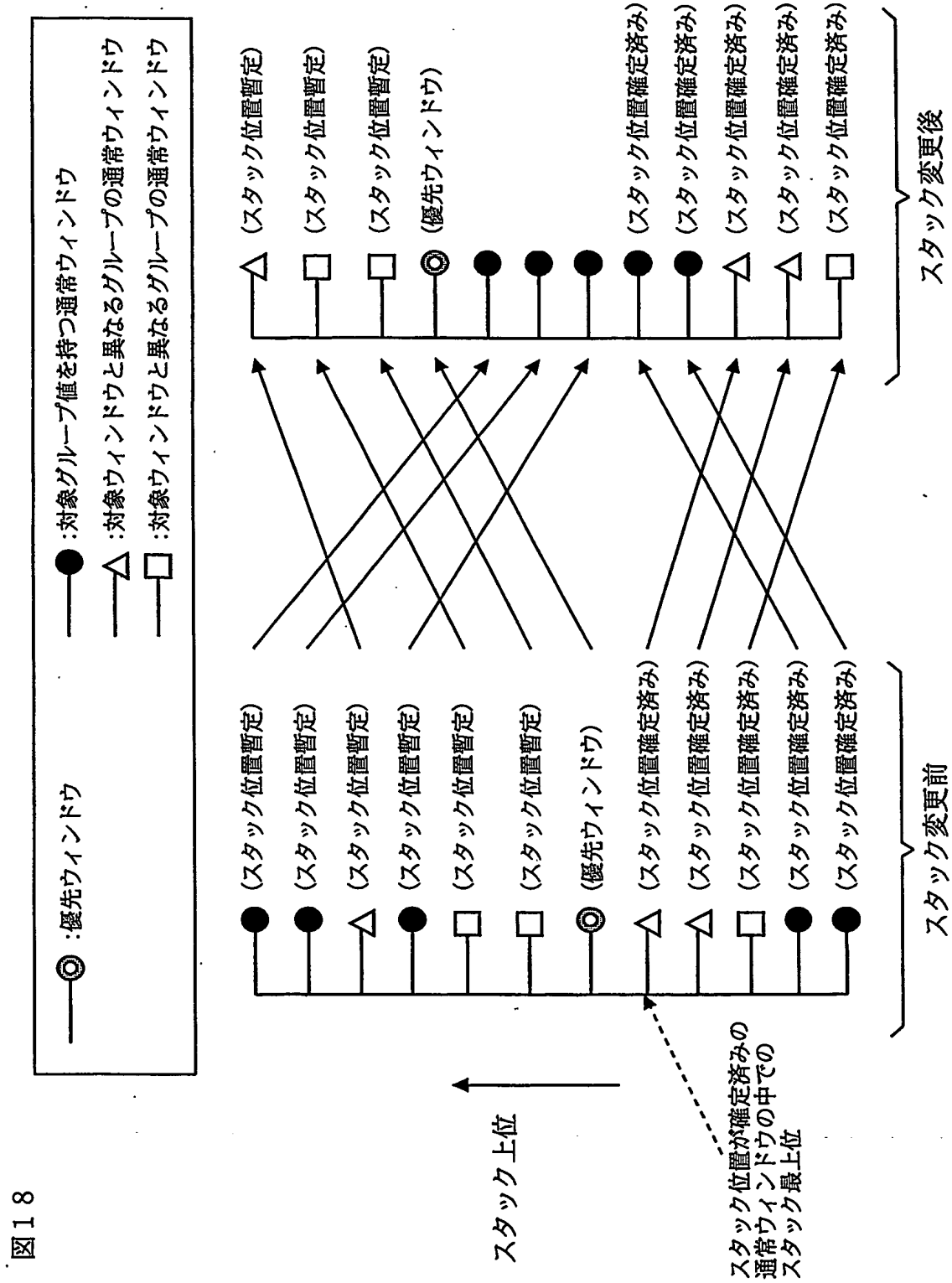

:優先ウィンドウ
:対象グループ値を持つ通常ウィンドウ
:対象ウィンドウと異なるグループの通常ウィンドウ
:対象ウィンドウと異なるグループの通常ウィンドウ
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(優先ウィンドウ)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
スタック上位
スタック位置が確定済みの
通常ウィンドウの中での
スタック最上位
スタック変更前
スタック変更後

図19

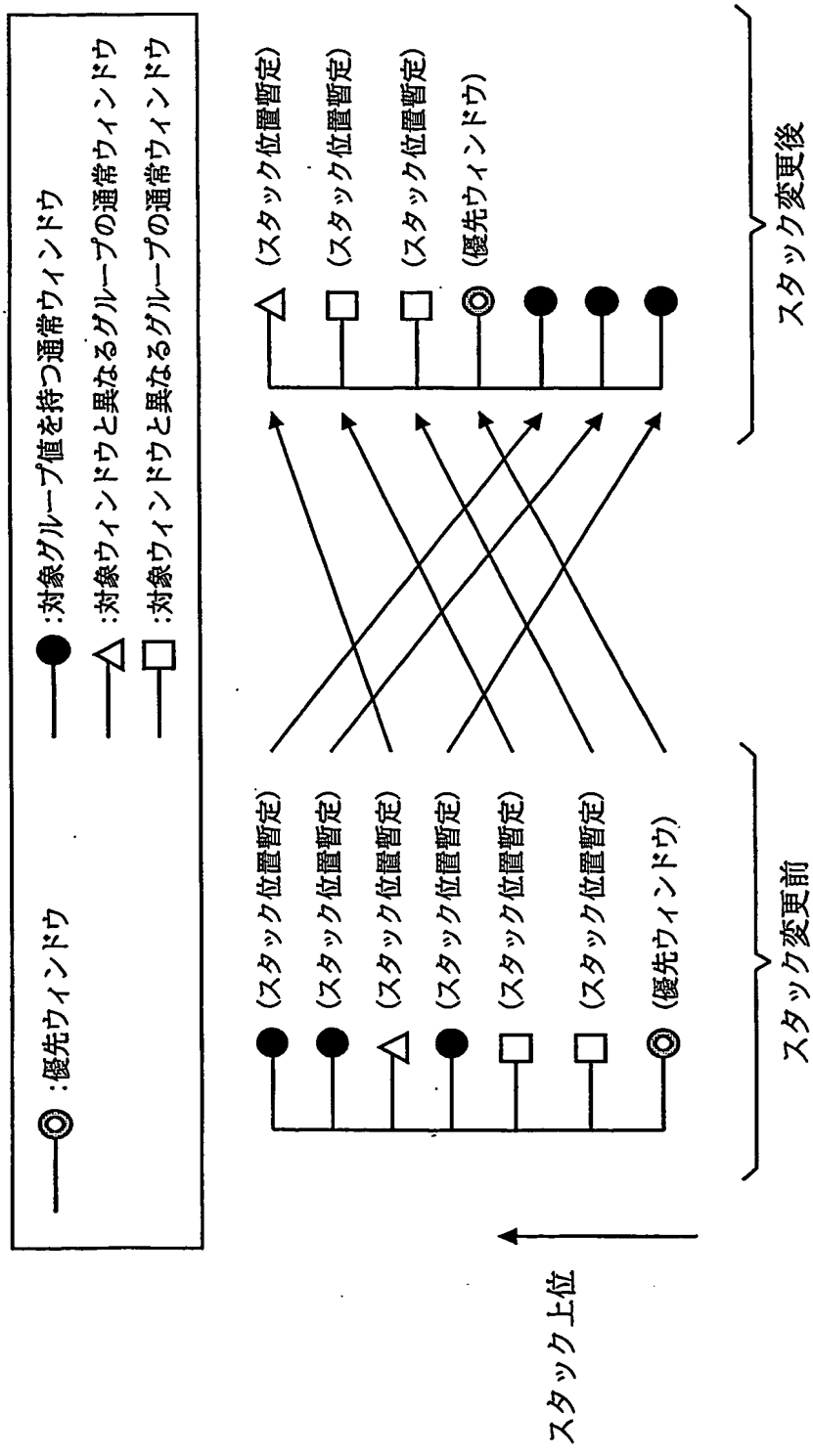

:優先ウィンドウ
:対象グループ値を持つ通常ウィンドウ
:対象ウィンドウと異なるグループの通常ウィンドウ
:対象ウィンドウと異なるグループの通常ウィンドウ
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(優先ウィンドウ)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(優先ウィンドウ)
スタック上位
スタック変更前
スタック変更後

図２０

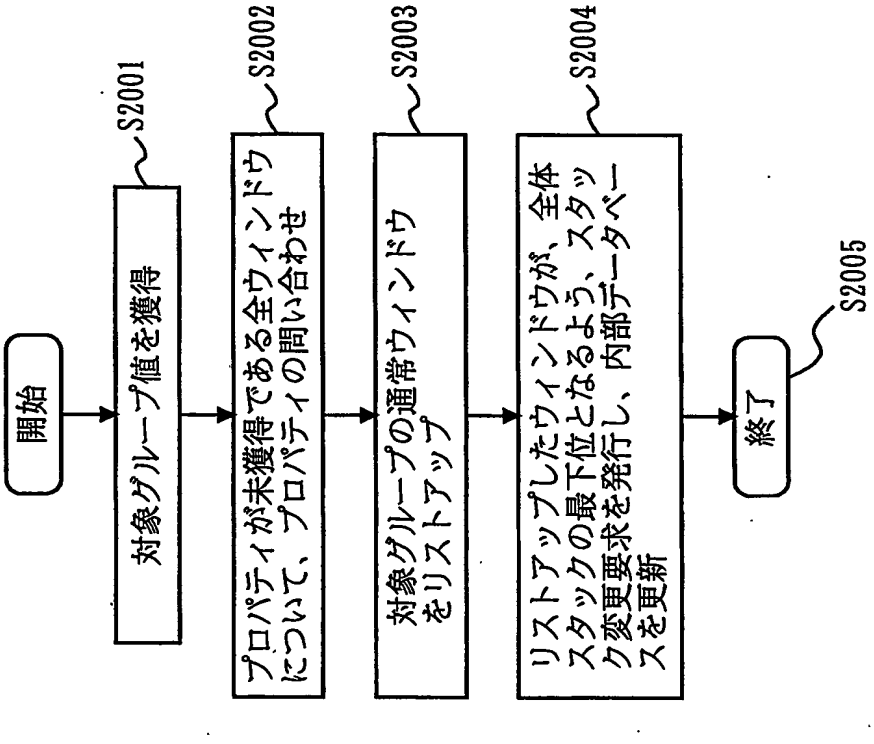

開始
対象グループ値を獲得
S2001
プロパティが未獲得である全ウィンドウについて、プロパティの問い合わせ
S2002
対象グループの通常ウィンドウをリストアップ
S2003
リストアップしたウィンドウが、全体スタックの最下位となるよう、スタック変更要求を発行し、内部データベースを更新
S2004
終了
S2005

図２１

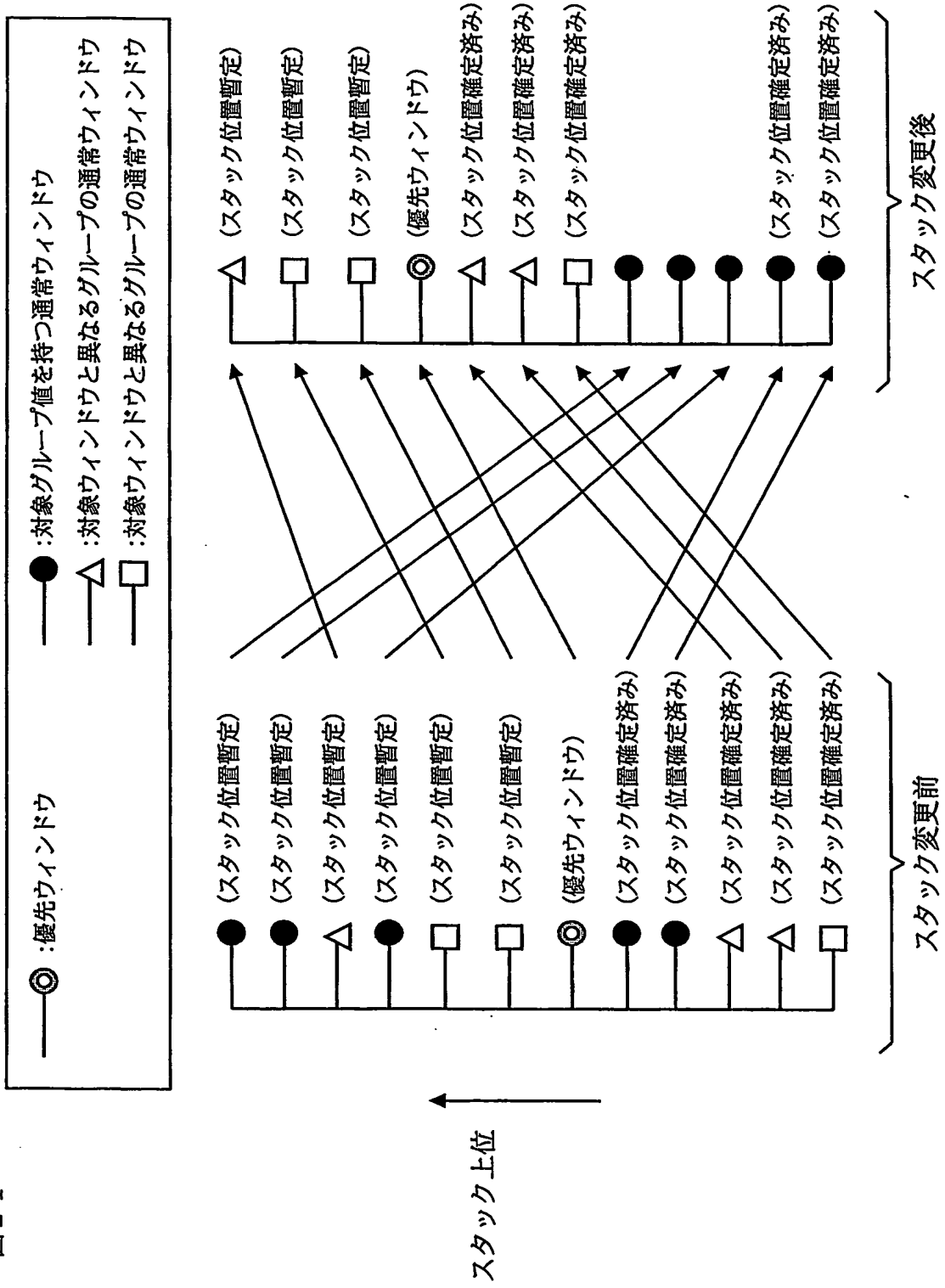

◎:優先ウィンドウ
●:対象グループ値を持つ通常ウィンドウ
△:対象ウィンドウと異なるグループの通常ウィンドウ
□:対象ウィンドウと異なるグループの通常ウィンドウ
スタック上位
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(優先ウィンドウ)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
スタック変更前
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(優先ウィンドウ)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
スタック変更後

図２２

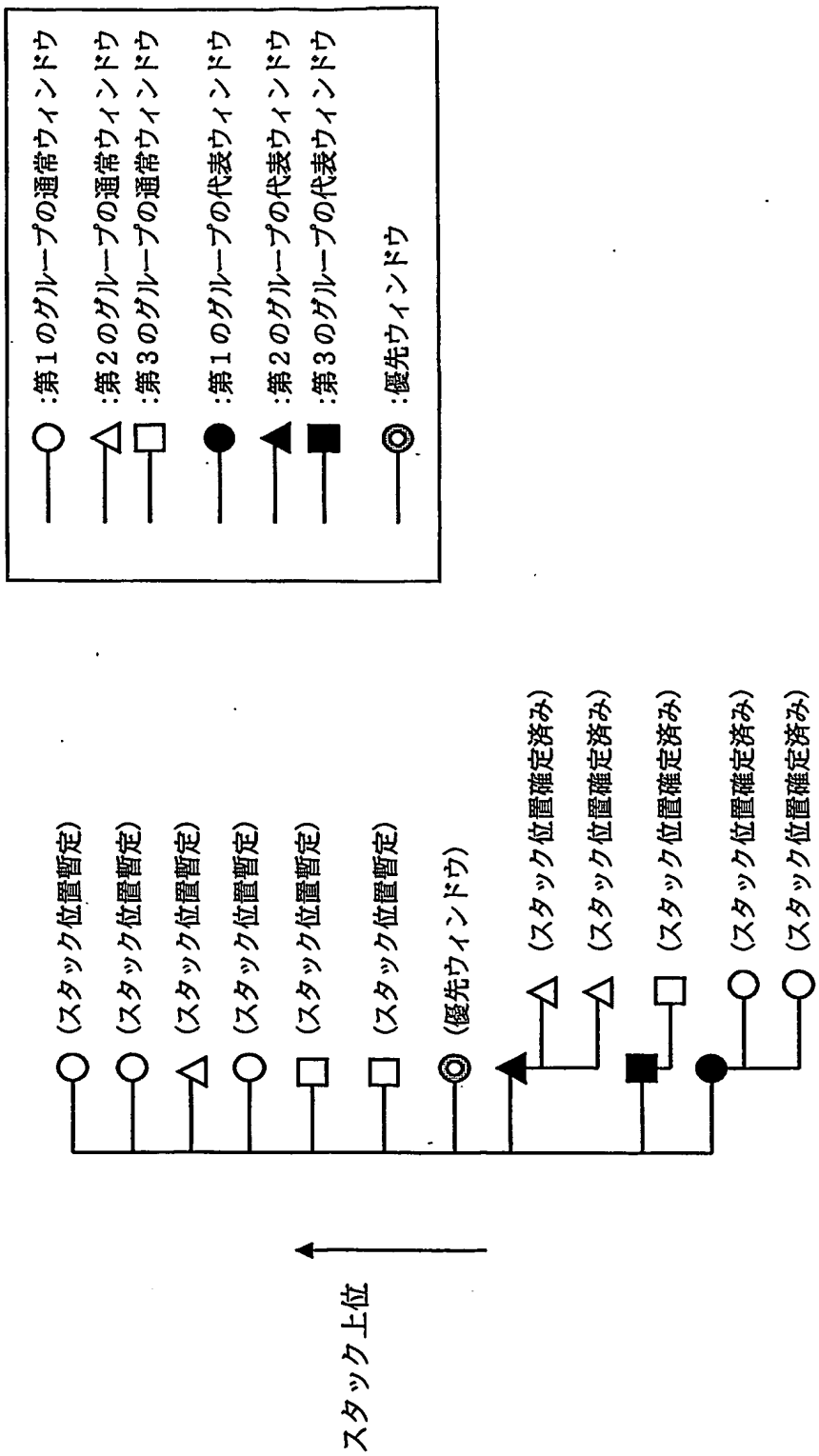
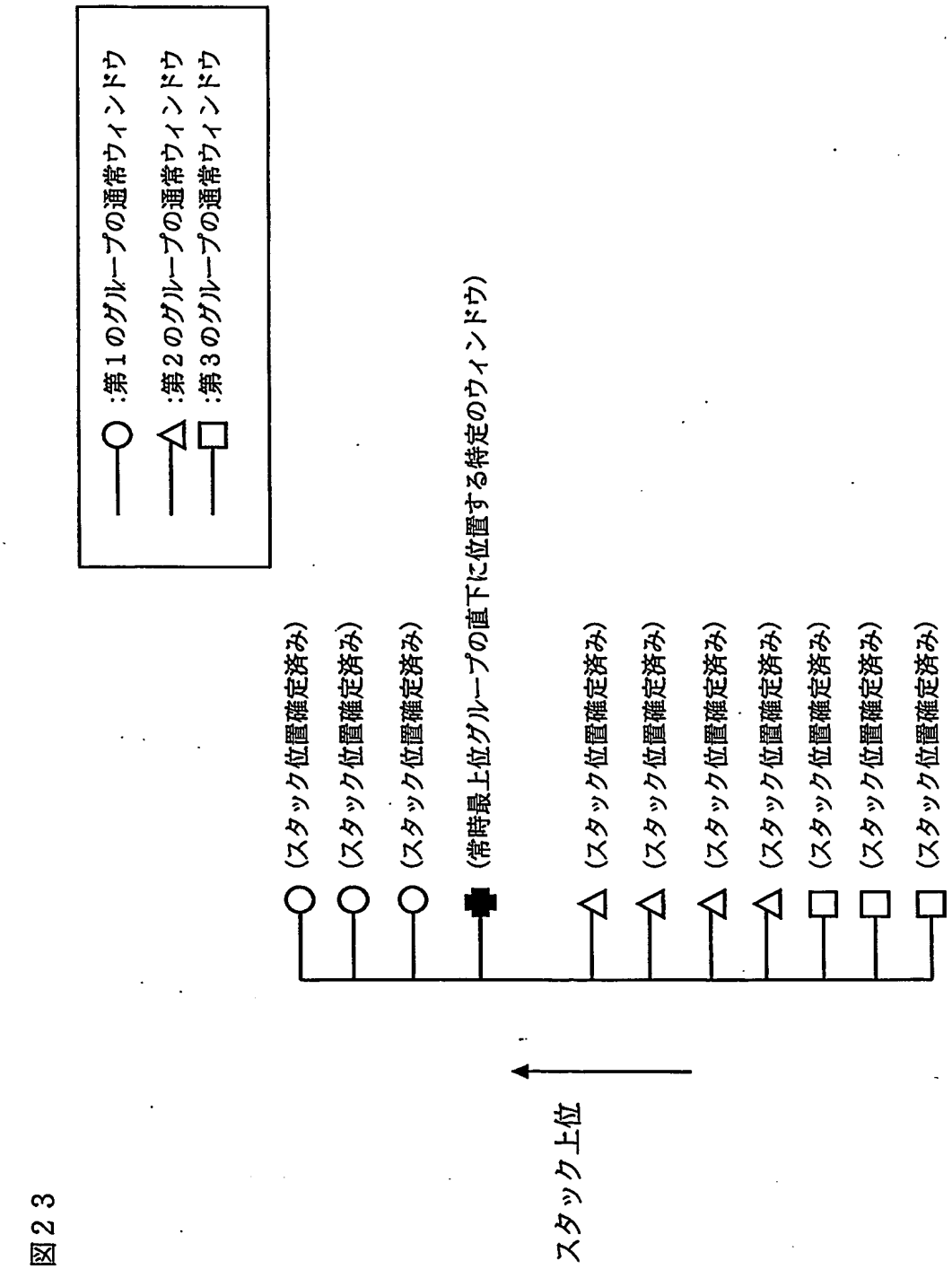

:第１のグループの通常ウィンドウ
:第２のグループの通常ウィンドウ
:第３のグループの通常ウィンドウ
:第１のグループの代表ウィンドウ
:第２のグループの代表ウィンドウ
:第３のグループの代表ウィンドウ
:優先ウィンドウ
スタック上位
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(優先ウィンドウ)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)

図２３


:第１のグループの通常ウィンドウ
:第２のグループの通常ウィンドウ
:第３のグループの通常ウィンドウ
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
(常時最上位グループの直下に位置する特定のウィンドウ)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
スタック上位

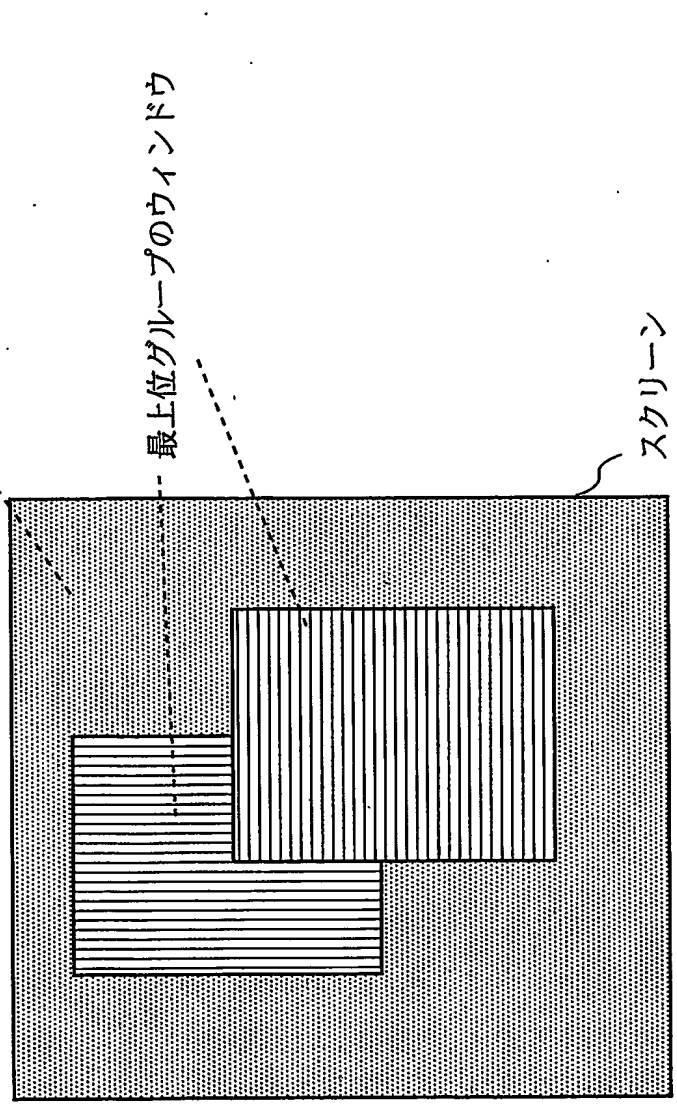


図２４

図２５

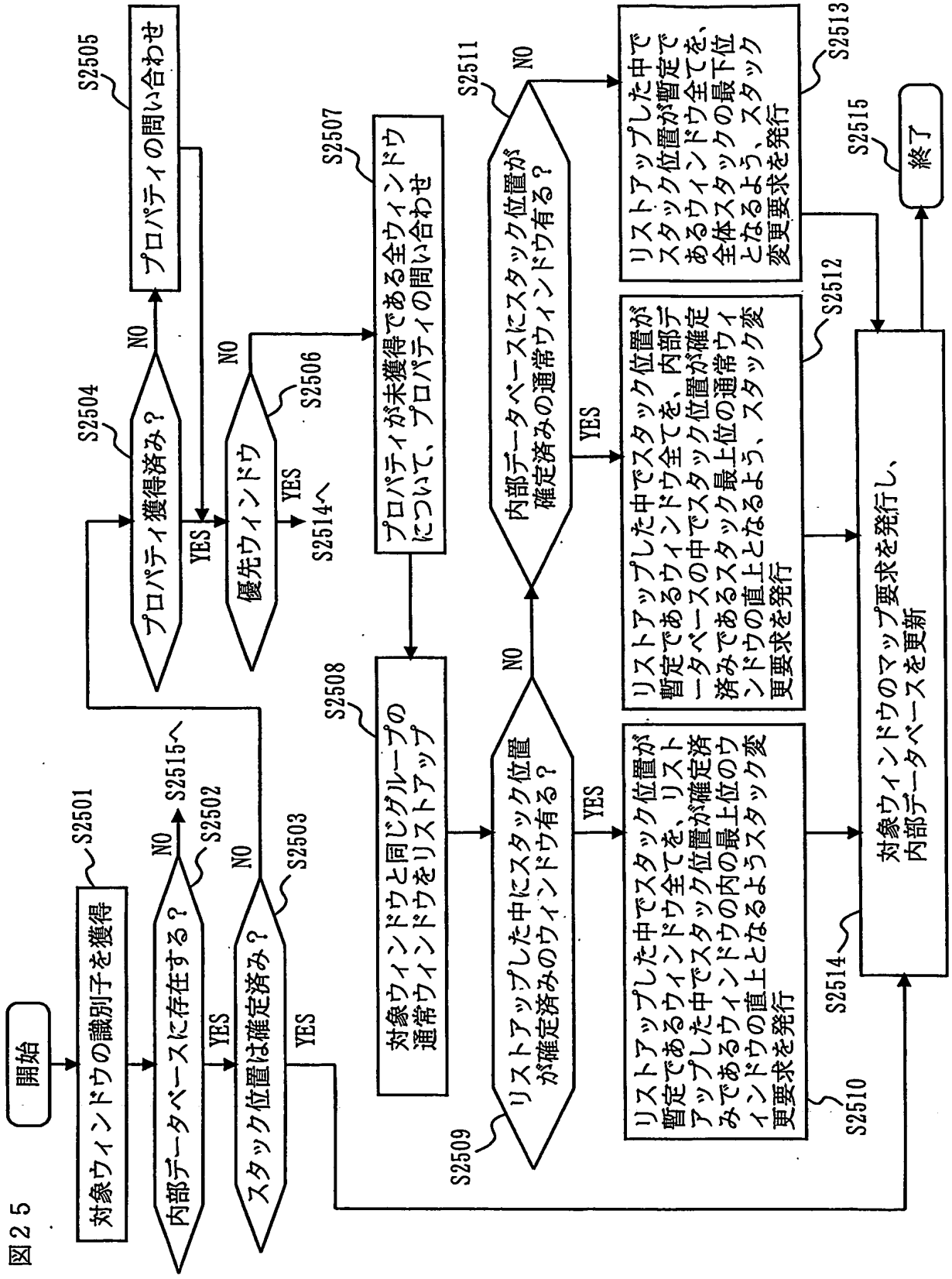

開始
対象ウィンドウの識別子を獲得
S2501
内部データベースに存在する？
NO
S2515へ
S2502
YES
スタック位置は確定済み？
NO
S2503
YES
プロパティ獲得済み？
NO
S2504
YES
プロパティの問い合わせ
S2505
優先ウィンドウ
NO
S2506
YES
S2514へ
プロパティが未獲得である全ウィンドウについて、プロパティの問い合わせ
S2507
対象ウィンドウと同じグループの通常ウィンドウをリストアップ
S2508
リストアップした中にスタック位置が確定済みのウィンドウ有る？
NO
S2509
YES
内部データベースにスタック位置が確定済みの通常ウィンドウ有る？
NO
S2511
YES
リストアップした中でスタック位置が暫定であるウィンドウ全てを、リストアップした中でスタック位置が確定済みであるウィンドウの内の最上位のウィンドウの直上となるようスタック変更要求を発行
S2510
リストアップした中でスタック位置が確定済みであるウィンドウの中でスタック最上位の通常ウィンドウの直上となるよう、スタック変更要求を発行
S2512
リストアップした中でスタック位置が暫定であるウィンドウ全てを、全体スタックの最下位となるよう、スタック変更要求を発行
S2513
S2514
対象ウィンドウのマップ要求を発行し、内部データベースを更新
S2515
終了

図２６

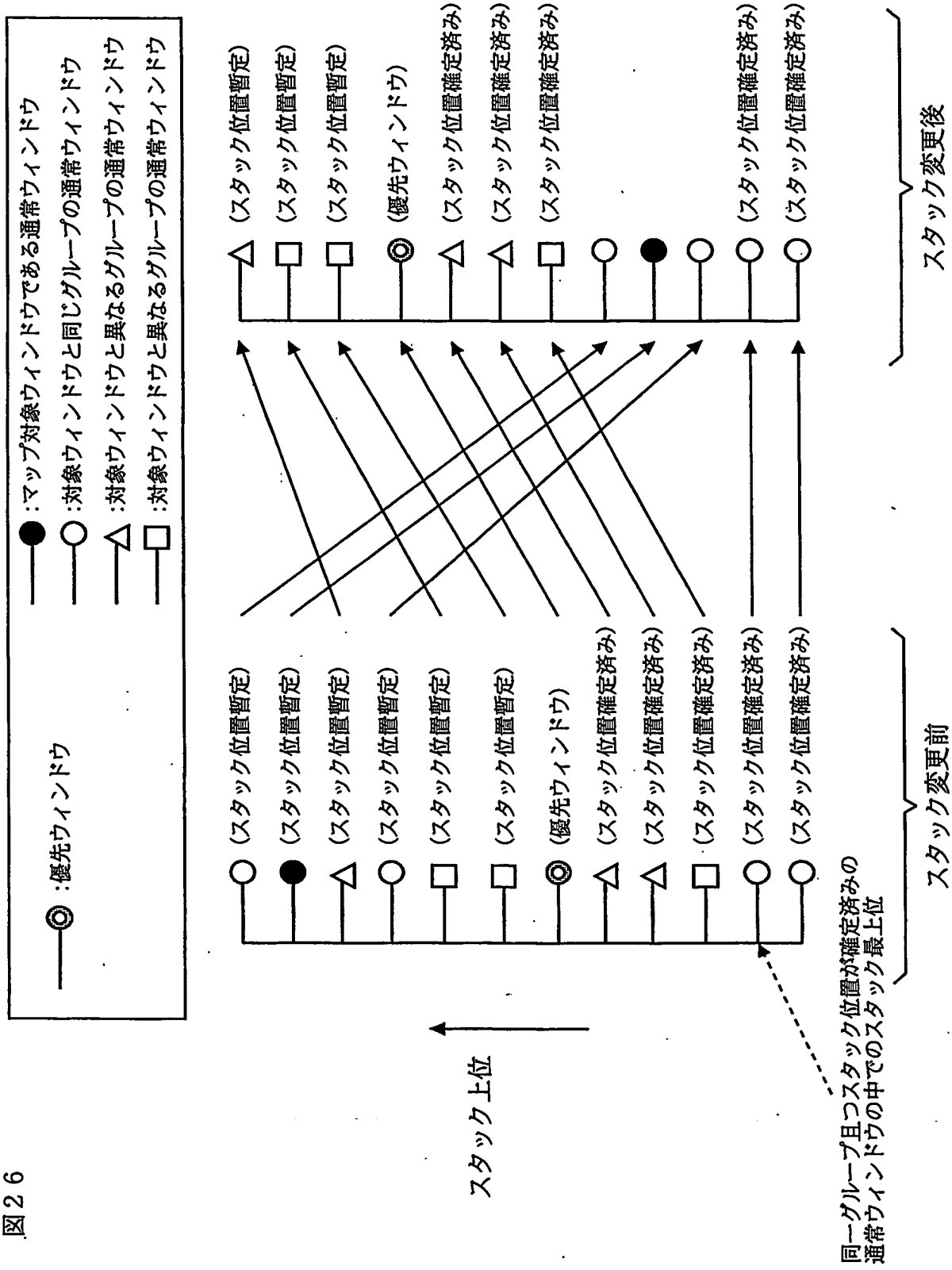

:マップ対象ウィンドウである通常ウィンドウ
:優先ウィンドウ
:対象ウィンドウと同じグループの通常ウィンドウ
:対象ウィンドウと異なるグループの通常ウィンドウ
:対象ウィンドウと異なるグループの通常ウィンドウ
スタック上位
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(優先ウィンドウ)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(優先ウィンドウ)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
同一グループ且つスタック位置が確定済みの
通常ウィンドウの中でのスタック最上位
スタック変更前
スタック変更後

図27

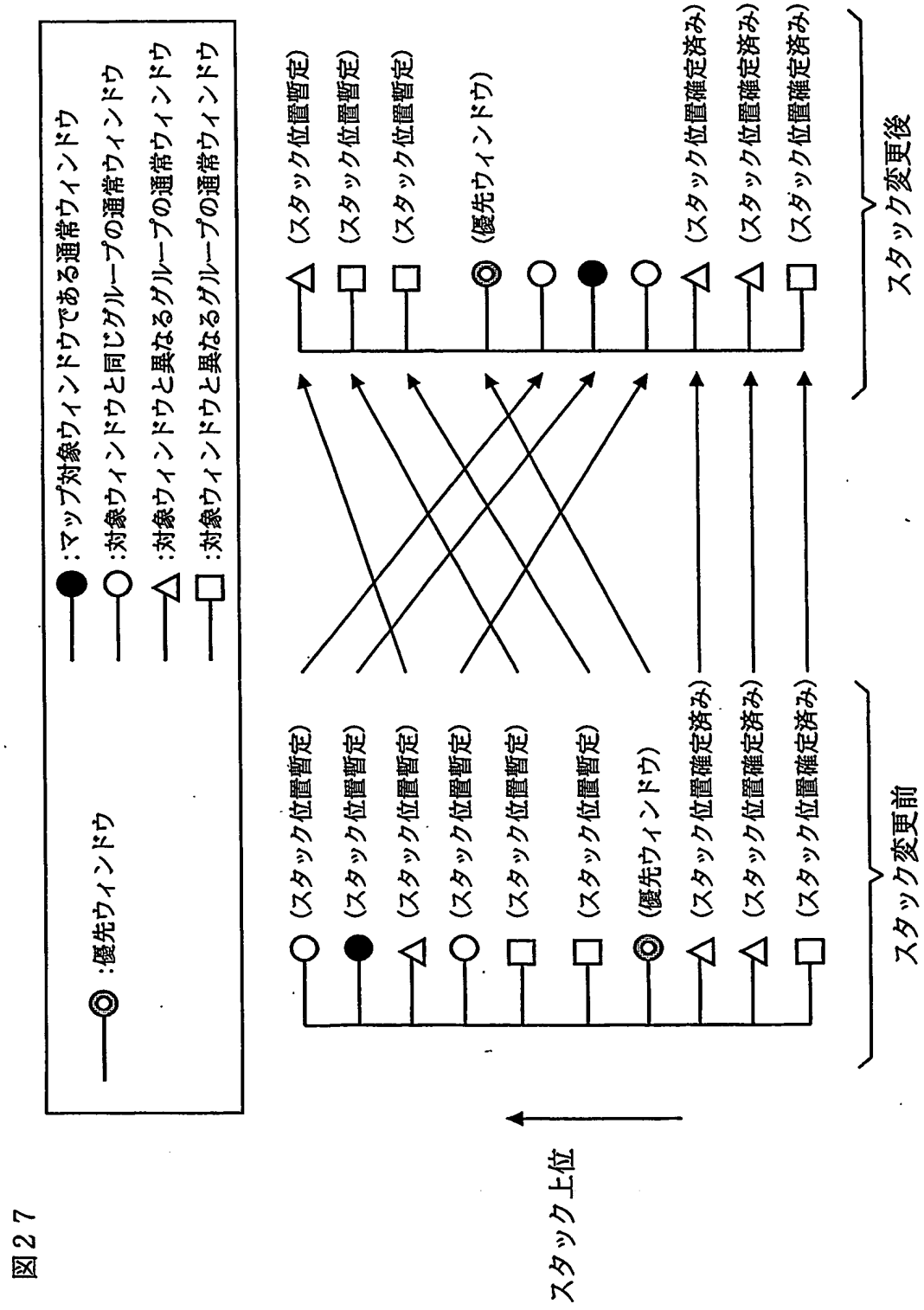
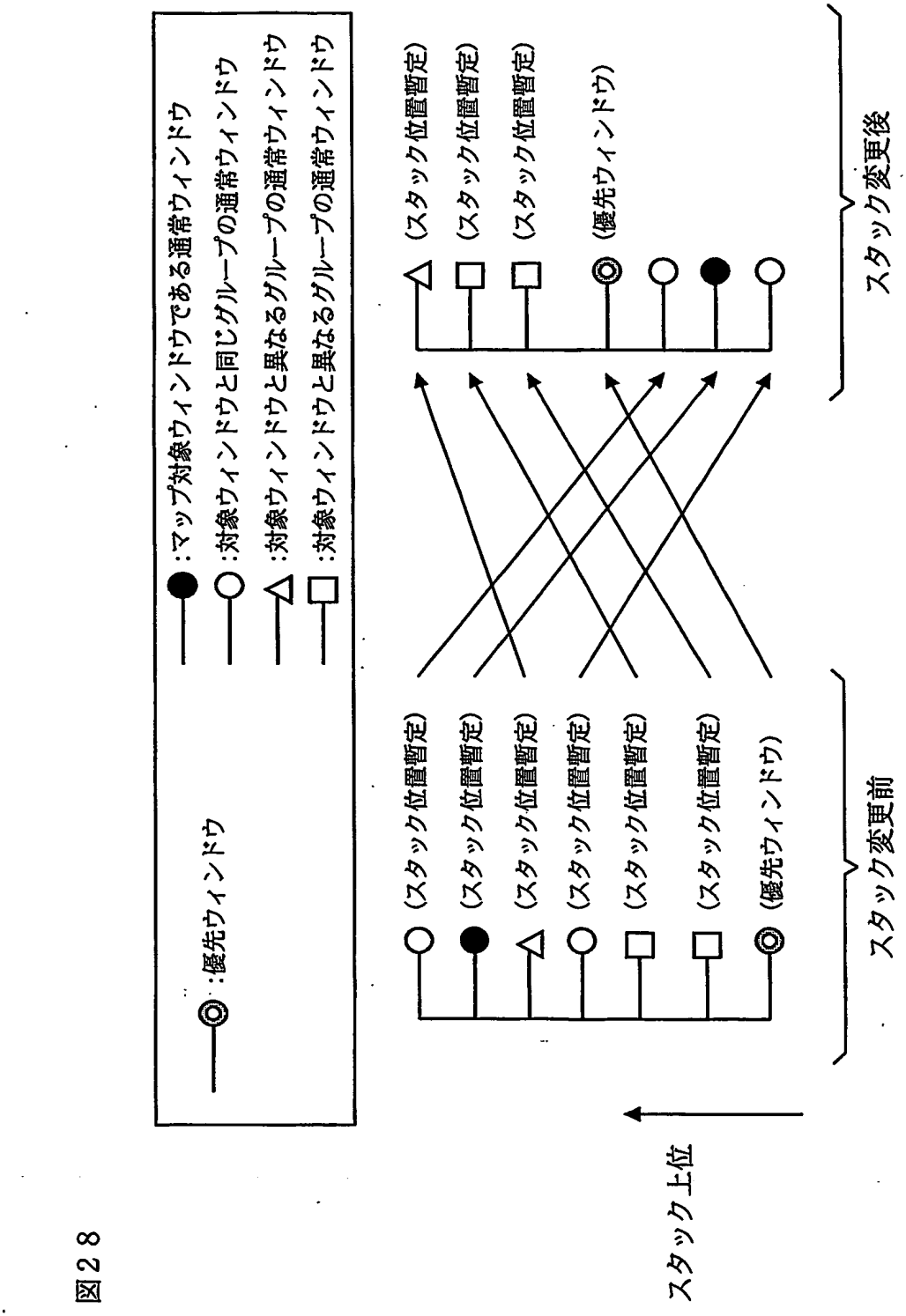

:マップ対象ウィンドウである通常ウィンドウ
:優先ウィンドウ
:対象ウィンドウと同じグループの通常ウィンドウ
:対象ウィンドウと異なるグループの通常ウィンドウ
:対象ウィンドウと異なるグループの通常ウィンドウ
スタック上位
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(優先ウィンドウ)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
スタック変更前
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(優先ウィンドウ)
(スタック位置確定済み)
(スタック位置確定済み)
(スタック位置確定済み)
スタック変更後

図２８


●:マップ対象ウィンドウである通常ウィンドウ
○:対象ウィンドウと同じグループの通常ウィンドウ
△:対象ウィンドウと異なるグループの通常ウィンドウ
□:対象ウィンドウと異なるグループの通常ウィンドウ
◎:優先ウィンドウ
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(優先ウィンドウ)
スタック上位
スタック変更前
(スタック位置暫定)
(スタック位置暫定)
(スタック位置暫定)
(優先ウィンドウ)
スタック変更後

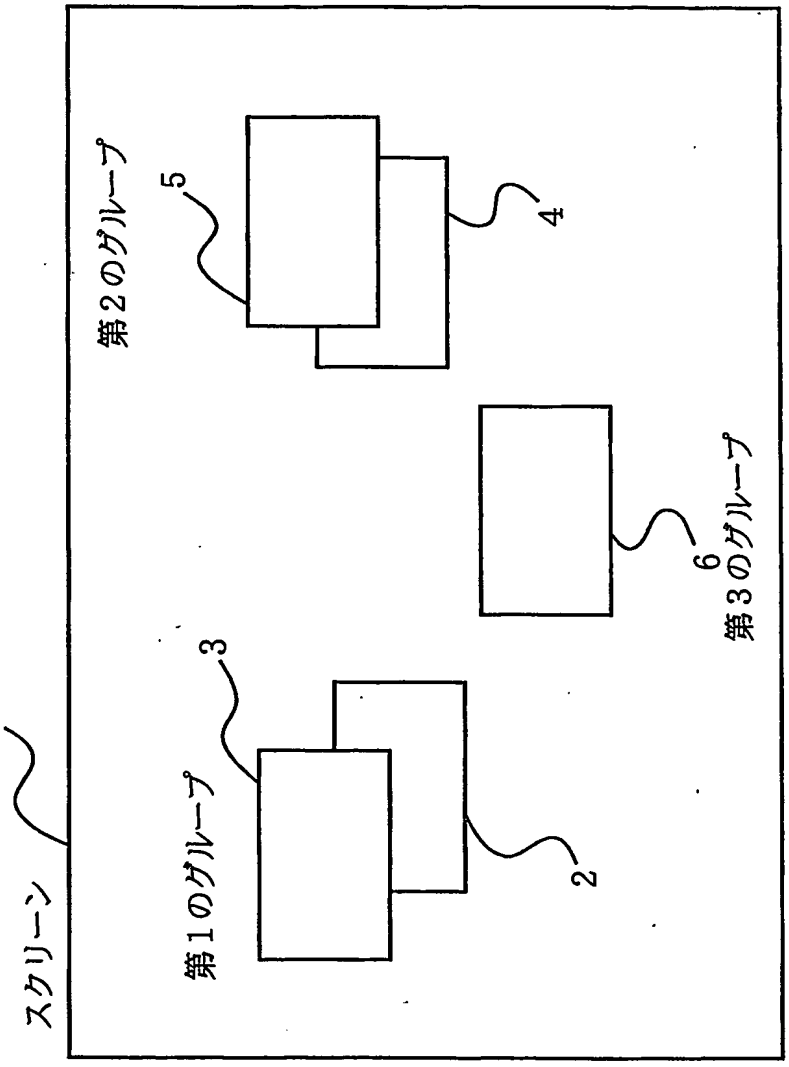


図29

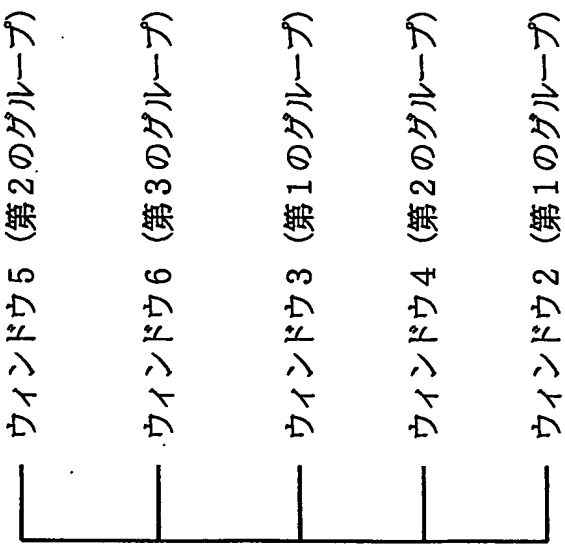

ウィンドウ５（第２のグループ）
ウィンドウ６（第３のグループ）
ウィンドウ３（第１のグループ）
ウィンドウ４（第２のグループ）
ウィンドウ２（第１のグループ）
スタック上位

図３０

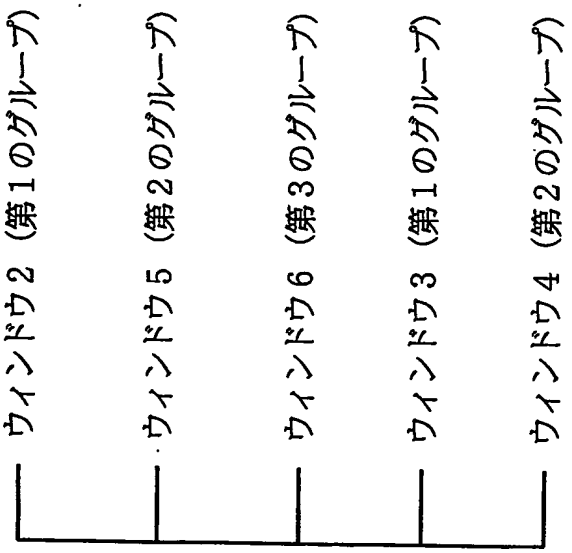

ウィンドウ２（第１のグループ）
ウィンドウ５（第２のグループ）
ウィンドウ６（第３のグループ）
ウィンドウ３（第１のグループ）
ウィンドウ４（第２のグループ）
スタック上位

図３１

INTERNATIONAL SEARCH REPORT

International application No.
PCT/JP2004/004414

A. CLASSIFICATION OF SUBJECT MATTER
Int.Cl[7] G06F3/14, G09G5/14

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
Int.Cl[7] G06F3/14, G06F3/00, G09G5/14

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Jitsuyo Shinan Koho	1922-1996	Toroku Jitsuyo Shinan Koho	1994-2004
Kokai Jitsuyo Shinan Koho	1971-2004	Jitsuyo Shinan Toroku Koho	1996-2004

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	JP 02-244323 A (Fujitsu Ltd.), 28 September, 1990 (28.09.90), Full text; all drawings	1,2,4-8, 11-16
Y	Full text; all drawings (Family: none)	3,9
X	JP 2001-060134 A (Hitachi, Ltd.), 06 March, 2001 (06.03.01), Full text; all drawings	1,2,4-8, 11-16
Y	Full text; all drawings (Family: none)	3,9

☒ Further documents are listed in the continuation of Box C. ☐ See patent family annex.

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier application or patent but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search 23 June, 2004 (23.06.04)	Date of mailing of the international search report 06 July, 2004 (06.07.04)
Name and mailing address of the ISA/ Japanese Patent Office Facsimile No.	Authorized officer Telephone No.

Form PCT/ISA/210 (second sheet) (January 2004)

INTERNATIONAL SEARCH REPORT

International application No.
PCT/JP2004/004414

C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
	JP 05-282119 A (Nippon Telegraph And Telephone Corp.), 29 October, 1993 (29.10.93),	
X	Full text; all drawings	1,2,4-8, 10-16
Y	Full text; all drawings (Family: none)	3,9
	JP 63-064121 A (Casio Computer Co., Ltd.), 22 March, 1988 (22.03.88),	
X	Full text; all drawings	1-8,11-16
Y	Full text; all drawings (Family: none)	3,9
Y	JP 03-080323 A (Fujitsu Ltd.), 05 April, 1991 (05.04.91), Full text; all drawings (Family: none)	3
Y	JP 06-044031 A (NEC Corp.), 18 February, 1994 (18.02.94), Full text; all drawings (Family: none)	9

国際調査報告

国際出願番号　ＰＣＴ／ＪＰ２００４／００４４１４

A.　発明の属する分野の分類（国際特許分類（ＩＰＣ））

Ｉｎｔ．Ｃｌ⁷　Ｇ０６Ｆ　３／１４，　Ｇ０９Ｇ　５／１４

B.　調査を行った分野

調査を行った最小限資料（国際特許分類（ＩＰＣ））

Ｉｎｔ．Ｃｌ⁷　Ｇ０６Ｆ　３／１４，　Ｇ０６Ｆ　３／００，　Ｇ０９Ｇ　５／１４

最小限資料以外の資料で調査を行った分野に含まれるもの

日本国実用新案公報　１９２２－１９９６年
日本国公開実用新案公報　１９７１－２００４年
日本国登録実用新案公報　１９９４－２００４年
日本国実用新案登録公報　１９９６－２００４年

国際調査で使用した電子データベース（データベースの名称、調査に使用した用語）

C.　関連すると認められる文献

引用文献のカテゴリー＊	引用文献名　及び一部の箇所が関連するときは、その関連する箇所の表示	関連する請求の範囲の番号
	ＪＰ　０２－２４４３２３　Ａ　（富士通株式会社）　１９９０．０９．２８	
Ｘ	全文，全図	1, 2, 4－8, 11－16
Ｙ	全文，全図 （ファミリーなし）	3, 9
	ＪＰ　２００１－０６０１３４　Ａ　（株式会社日立製作所）　２００１．０３．０６	
Ｘ	全文，全図	1, 2, 4－8, 11－16
Ｙ	全文，全図 （ファミリーなし）	3, 9

☒ Ｃ欄の続きにも文献が列挙されている。　☐ パテントファミリーに関する別紙を参照。

＊　引用文献のカテゴリー
「Ａ」特に関連のある文献ではなく、一般的技術水準を示すもの
「Ｅ」国際出願日前の出願または特許であるが、国際出願日以後に公表されたもの
「Ｌ」優先権主張に疑義を提起する文献又は他の文献の発行日若しくは他の特別な理由を確立するために引用する文献（理由を付す）
「Ｏ」口頭による開示、使用、展示等に言及する文献
「Ｐ」国際出願日前で、かつ優先権の主張の基礎となる出願の日の後に公表された文献
「Ｔ」国際出願日又は優先日後に公表された文献であって出願と矛盾するものではなく、発明の原理又は理論の理解のために引用するもの
「Ｘ」特に関連のある文献であって、当該文献のみで発明の新規性又は進歩性がないと考えられるもの
「Ｙ」特に関連のある文献であって、当該文献と他の１以上の文献との、当業者にとって自明である組合せによって進歩性がないと考えられるもの
「＆」同一パテントファミリー文献

国際調査を完了した日　２３．０６．２００４	国際調査報告の発送日　06.7.2004
国際調査機関の名称及びあて先 日本国特許庁（ＩＳＡ／ＪＰ） 郵便番号１００－８９１５ 東京都千代田区霞が関三丁目４番３号	特許庁審査官（権限のある職員）　５Ｅ　９７４３ 馬場　慎 電話番号　０３－３５８１－１１０１　内線　３５２０

様式ＰＣＴ／ＩＳＡ／２１０（第２ページ）（２００４年１月）

国際調査報告

国際出願番号　ＰＣＴ／ＪＰ２００４／００４４１４

Ｃ（続き）．　関連すると認められる文献

引用文献のカテゴリー＊	引用文献名　及び一部の箇所が関連するときは、その関連する箇所の表示	関連する請求の範囲の番号
	ＪＰ　０５－２８２１１９　Ａ　（日本電信電話株式会社）　１９９３．１０．２９	
Ｘ	全文，全図	1，2，4－8，10－16
Ｙ	全文，全図	3，9
	（ファミリーなし）	
	ＪＰ　６３－０６４１２１　Ａ　（カシオ計算機株式会社）　１９８８．０３．２２	
Ｘ	全文，全図	１－８，１１－１６
Ｙ	全文，全図	3，9
	（ファミリーなし）	
Ｙ	ＪＰ　０３－０８０３２３　Ａ　（富士通株式会社）　１９９１．０４．０５	3
	全文，全図	
	（ファミリーなし）	
Ｙ	ＪＰ　０６－０４４０３１　Ａ　（日本電気株式会社）　１９９４．０２．１８	9
	全文，全図	
	（ファミリーなし）	